الجامعة الافتراضية السورية
SYRIAN VIRTUAL UNIVERSITY

ماتلاب للحوسبة العددية

المهندس سيمون طربوش

ISSN: 2617-989X

Books & Refrences

ماتلاب للحوسبة العددية

المهندس سيمون طربوش

من منشورات الجامعة الافتراضية السورية

الجمهورية العربية السورية 2020

MATLAB for Numerical Computing

Eng. **Simon Tarbouche**

Publications of the Syrian Virtual University (SVU)

Syrian Arab Republic, 2020

الفهرس

الفصل الأول: مدخل إلى بيئة MATLAB®
(مفاهيم عامة)
Introduction to MATLAB® Environment
(General Concepts)

عنوان الموضوع:

مدخل إلى بيئة MATLAB® (مفاهيم عامة)

Introduction to MATLAB® Environment (General Concepts)

الكلمات المفتاحية:

MATLAB®، MathWorks®، لغة برمجة عالية المستوى، Workspace، Command Window، Help، Documentation، Toolstrip، واجهة بيانية تخاطبية GUI، Simulink، Toolboxes.

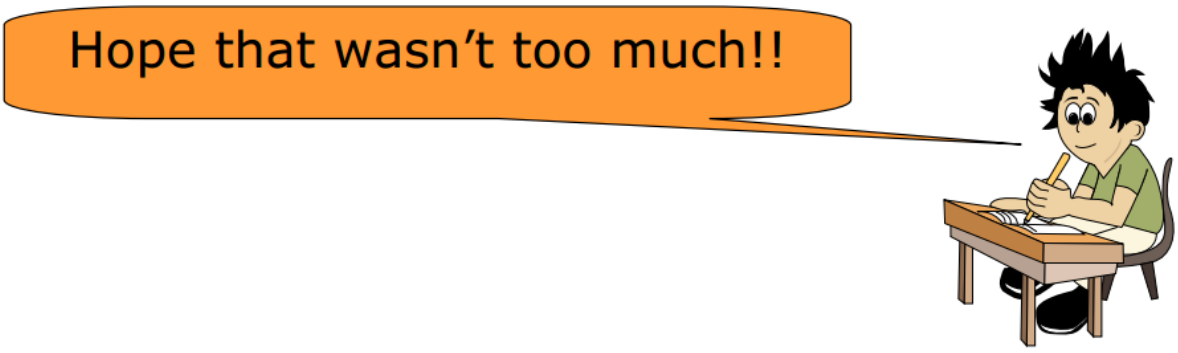
ملخص:

نقدم في هذا الفصل تعريف ببرمجية MATLAB® وأجزائها، حيث سنتعرف على لغة البرمجة MATLAB® والاستخدامات العديدة لهذه البرمجية في تطبيقات ومجالات متنوعة كهندسة الاتصالات والتحكم، وسنتعرف أيضاً على الميزات الهامة والمختلفة التي تتمتع بها برمجية MATLAB®. ثم سننتقل إلى التعرف على الواجهات البيانية ضمن البرمجية، وطريقة استخدام كل منها.

أهداف تعليمية:

يتعرف الطالب في هذا الفصل على:

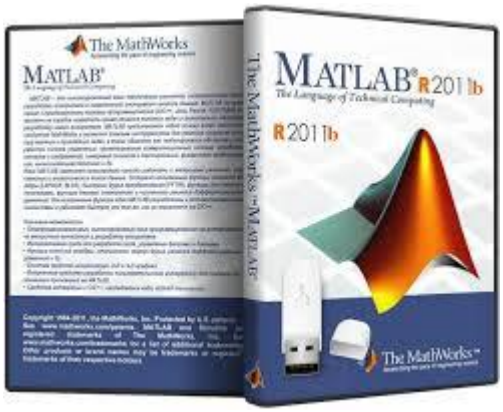
- مقدمة عن لغة البرمجة MATLAB®
- مكونات البيئة
- استخدامات MATLAB® في المجالات المتنوعة وميزاته
- استخدام HELP الموجودة ضمن البرمجية وهو أكثر الأمور أهميةً حتى نهاية المقرر.

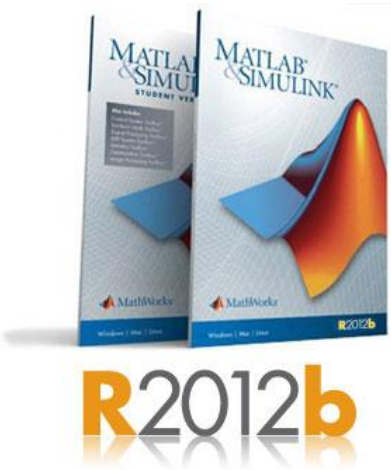


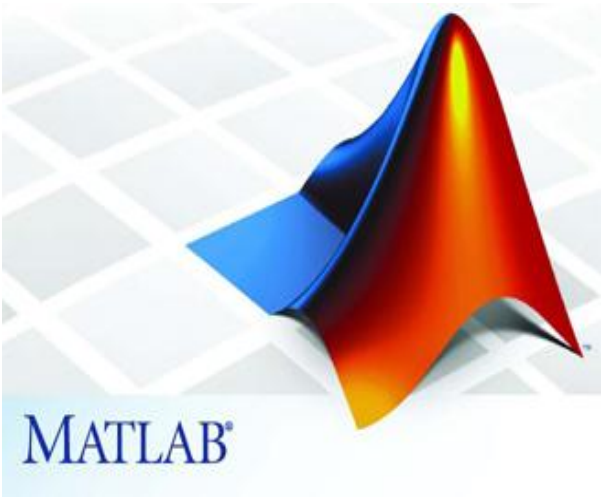
- التعامل مع الواجهة التخاطبية للبرمجية وفهم مكوناتها، واستخدامات كل منها.

ماهو MATLAB وما هي استخداماته

1. التعرف على MATLAB







MATLAB، اختصار لـMATrix LABoratory أي مختبر المصفوفات، وهي إحدى لغات البرمجة عالية المستوى، قامت شركة MathWorks® بتطوير بيئة برمجية أطلقت عليها اسم MATLAB أيضاً، بهدف برمجة وتطوير التطبيقات اعتماداً على لغة MATLAB؛ تتميز هذه البيئة بأنها تفاعلية وسهلة الاستعمال حيث يتم صياغة المشاكل والحلول باستخدام الصيغ والرموز الرياضية، تستخدم هذه البرمجية من قبل الملايين من المهندسين والعلماء في جميع أنحاء العالم.

2. لمحة تاريخية عن البرمجية

نقدم في هذه الفقرة لمحة تاريخية يفضل ان يتم عرضها على شكل تعدادات متلاحقة باستخدام bullets الموجودة ضمن word or powerpoint اي على الشكل

- قام Cleve Moler، المسؤول عن قسم علوم الحاسب في جامعة New Mexico، بالبدء بتطوير MATLAB في الفترة بين 1970-1980؛
- وقد كان **الهدف** من بناء لغة البرمجة وتصميم البرمجية هو **منح الطلاب القدرة على التعامل مع برمجيات مبنية بلغة Fortran دون الحاجة لتعلم لغة Fortran**
- وقد لاقت الفكرة نجاحاً كبيراً وأخذت بالانتشار إلى العديد من الجامعات، كما أنها راعت انتباه العاملين في مجال الرياضيات التطبيقية.
- بعد ذلك قام المهندس Jack Little بلقاء Cleve Moler وأدرك قيمة هذه البرمجية من الناحية التطبيقية والتجارية؛ لذلك قام المهندس جاك مع Cleve Moler و Steve Bangert بإنشاء شركة Mathwork® في عام 1984 وإعادة كتابة MATLAB بلغة C وإضافة العديد من المكاتب Libraries التي تحتوي على عدد كبير من التوابع الرياضية بهدف تسهيل عمل المستخدم وزيادة كفاءة البرمجية،
- ومنذ ذلك الحين وحتى الآن يتم تطوير البرمجية وإضافة مكاتب وأدوات Toolboxes تسهل العمل في مجالات علمية وهندسية مختلفة ومتنوعة حتى أصبحت البرمجية مستخدمة في أنظمة التحكم، معالجة الإشارة والاتصالات، معالجة الصورة والفيديو، الاختبار والقياس Test and Measurement.
- في عام 2004 أصبح عدد مستخدمي MATLAB حوالي مليون مستخدم موزعين بين الشركات الصناعية، الأبحاث العلمية و المجال الأكاديمي، في وقتنا الراهن يزيد عدد مستخدمي هذه البرمجية عن عشرة ملايين مستخدم، إضافةً لوجود أكثر من 1500 كتاب لتعليم MATLAB ب 27 لغة مختلفة.

3. نظام MATLAB

يتألف نظام MATLAB من أربعة أجزاء رئيسية هي

1. لغة البرمجة MATLAB (Prgramming Language MATLAB)

وهي لغة برمجة عالية المستوى تحتوي على control flow statement مثل if else, switch، توابع، بنى معطيات data structure، وميزات البرمجة غرضية التوجه من حيث التعامل مع صفوف classes ومفاهيم الوراثة والأغراض.

2. بيئة MATLAB (MATLAB Environment)

وهي البيئة المكونة من مجموعة من الواجهات والأدوات التي يتم العمل معها من قبل المستخدمين والمبرمجين، حيث تحتوي على حقول مخصصة لتعريف وتنظيم المتحولات، تصدير واستيراد معطيات، تطوير وتنفيذ برامج وتطبيقات.

3. Handle Graphics

وهو عبارة عن نظام الرسومات والبيانيات ضمن MATLAB، يحتوي على تعليمات لإنشاء رسومات ثنائية وثلاثية الأبعاد بهدف إظهار المعطيات، معالجة الصور، تحريك الرسوم وعرضها، إضافةً للعديد من الإضافات والخيارات لتوضيح الأشكال ككتابة تعليقات وعناوين وإضافة أسماء للمحاور؛ كما

يتضمن هذا النظام على أدوات مساعدة في بناء الواجهات البيانية التخاطبية GUI Graphical User Interface.

4. مكاتب التوابع الرياضية الخاصة بـMATLAB

وهي عبارة عن مجموعة واسعة من الخوارزميات الحسابية البسيطة (مثل الجمع، حساب جيب وتجيب الزاوية، العمليات العقدية) والمقعدة (مثل مقلوب مصفوفة، حساب القيم الذاتية لمصفوفة، توابع بيسل Bessel، تحويل فوريه السريع).

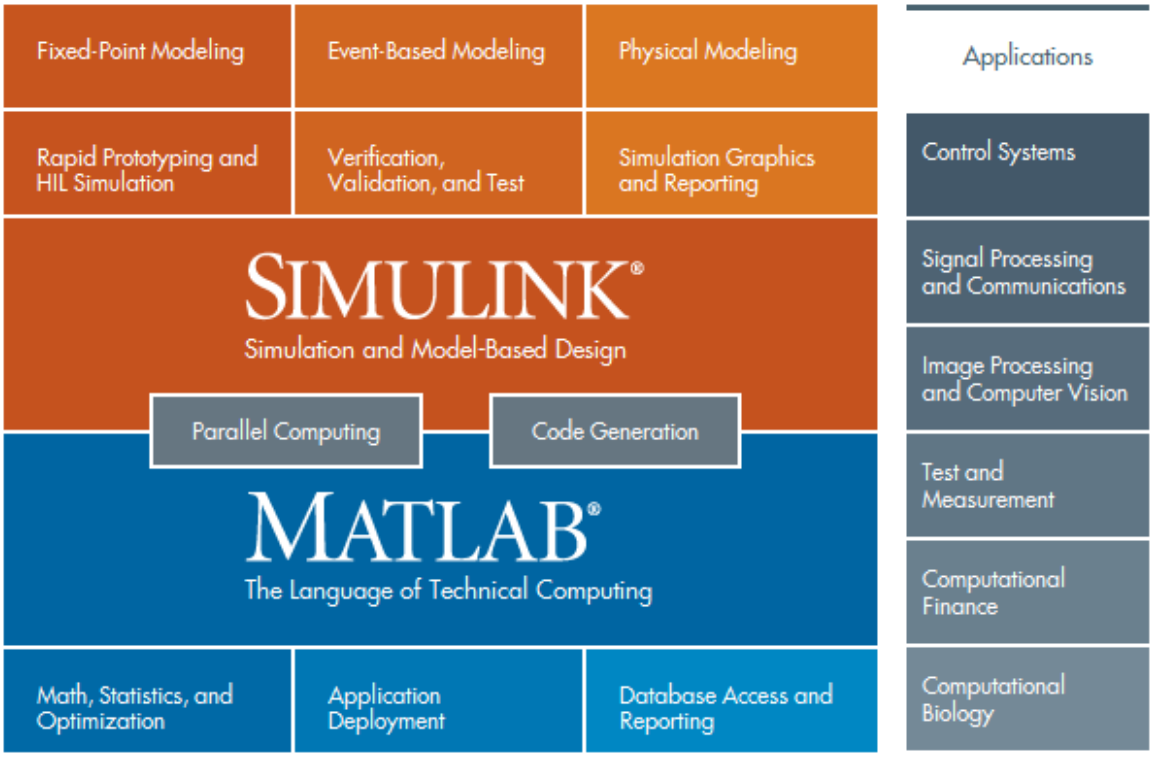
4. استخدامات MATLAB

تعتبر الصفيفة Array عنصر المعطيات الأساسي data element في هذه البرمجية، مما يتيح لنا إجراء عمليات مختلفة على المعطيات وذلك بغض النظر عن طريقة تمثيلها، كالمعطيات الممثلة بشكل مصفوفات Matrices حيث يمكن ضربها وجمعها وإجراء عدد كبير من العمليات عليها، كما تسمح لنا البرمجية بإجراء العمليات الحسابية، تحليل وإظهار المعطيات، رسم التوابع، تنجيز وتطوير الخوارزميات، إنشاء واجهات بيانية تخاطبية GUI Graphical User Interface ، التعامل مع تطبيقات مبنية اعتماداً على لغات برمجة مختلفة مثل C, C++, Java,C#, Fortran.

تقدم لنا هذه البرمجية أيضاً خيارات واسعة تفيد في الاسكتشاف، التصوّر Visualization[1] ومحاكاة الأفكار إضافةً إلى التطوير في مختلف الاختصاصات؛ نذكر منها الاتصالات، أنظمة التحكم، معالجة الإشارة والصورة.
إضافةً لذلك فهي تستخدم ضمن العديد من الجامعات والمعاهد العالمية في مجال التدريس كالجبر الخطي والتحليل العدّدي، البحث الأكاديمي، والنمذجة modeling.
تستخدم هذه البرمجية، عملياً، في بناء وتنجيز عدد كبير من التطبيقات والمشاريع في مجالات ومواضيع متنوعة؛ فمثلاً تتيح لنا نمذجة استهلاك الطاقة لبناء شبكات الطاقة الذكية، وتطوير خوارزميات التحكم، وتحليل بيانات الطقس لتصور أحول الطقس ومسار وشدة الأعاصير.
يبين لنا الشكل التالي منتجات شركة Mathwork والمواضيع المختلفة التي جرى مكاملتها ضمن برمجية MATLAB حيث تحتوي هذه البرمجية على قسم خاص بالبرمجة بلغة MATLAB، جزء Simulink من أجل المحاكاة، أدوات Toolboxes تهتم بعدد كبير من المجالات مثل الرياضيات و الإحصاء والأمثلة Optimization، أنظمة التحكم، معالجة الإشارة والاتصالات، الاختبارات والقياسات Test and Measurement.

[1] Visualization is the process of representing abstract business or scientific data as images or graphics that can aid in understanding the meaning of the data.

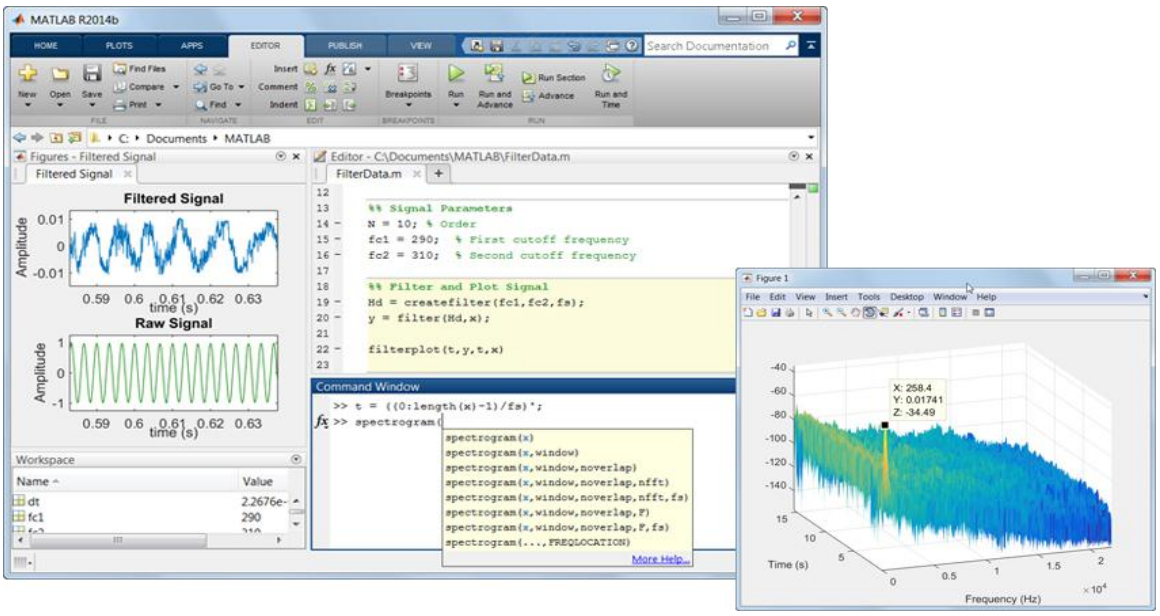
وهي عملية تمثيل المعطيات التجارية أوالعلمية المجردة باستخدام الصور أو الرسوم البيانية بحيث تساعد في فهم معنى ودلالة المعطيات


Fixed-Point Modeling
Event-Based Modeling
Physical Modeling
Rapid Prototyping and HIL Simulation
Verification, Validation, and Test
Simulation Graphics and Reporting
SIMULINK®
Simulation and Model-Based Design
Parallel Computing
Code Generation
MATLAB®
The Language of Technical Computing
Math, Statistics, and Optimization
Application Deployment
Database Access and Reporting
Applications
Control Systems
Signal Processing and Communications
Image Processing and Computer Vision
Test and Measurement
Computational Finance
Computational Biology

ميزات MATLAB، Features of MATLAB

1. ما هي الميزات

- لغة برمجة عالية المستوى تستخدم في الحسابات العدّدية، تطوير التطبيقات و Visualization.
- بيئة تفاعلية تستخدم في البحث واستخلاص النتائج، حل المشاكل، تصميم وتنفيذ خوارزميات مختلفة.
- تشتمل على توابع رياضية تفيد في مجالات الجبر الخطي، الإحصاء، تحليل فورييه Fourier Analysis، الترشيح Filtering، أمثّلة الخوارزميات Algorithms Optimization، التكاملات العدّدية، وحل المعادلات التفاضلية الجزئية Differential Equations.
- رسومات بيانية من ضمن البرمجية (built-in graphics) تفيد في إظهار المعطيات، إضافةً لأدوات Toolboxes تستخدم في إنشاء أشكال ومنحنيات تتناسب مع متطلبات المستخدم.
- أدوات تستخدم في تحسين نوعية الرماز code من حيث زمن التنفيذ، والحصول على أداء أفضل.
- واجهات بيانية GUI يقوم المستخدم ببنائها تحتوي على خيارات تصميمية متنوعة تتناسب مع التطبيق.
- توابع تفيد في مكاملة الخورازميات التي تم برمجتها باستخدام MATLAB مع تطبيقات ولغات برمجة مختلفة مثل C، JAVA، C#، .NET، و Microsoft Excel®.



عنوان الشكل: تحليل وإظهار المعطيات باستخدام برمجية MATLAB. إن بيئة MATLAB تتيح للمستخدم كتابة الرمازات البرمجية والبرامج وتطوير الخوارزميات والتطبيقات.

2. الحسابات العدّدية Numeric Computation

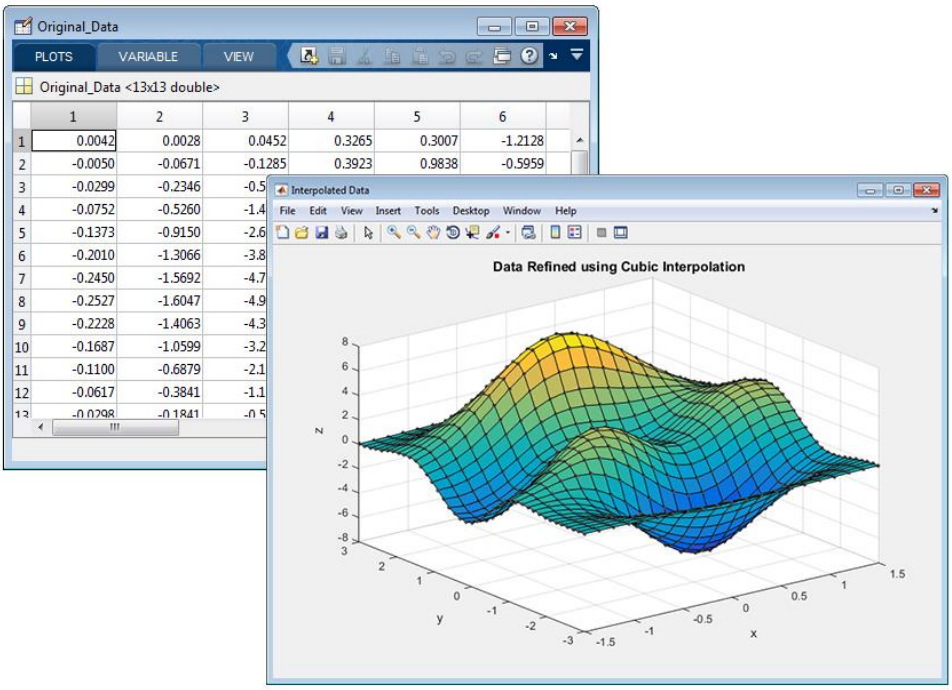
يؤمن لنا MATLAB مجموعة كبيرة من طرائق الحساب العدّدي تستخدم في معالجة المعطيات، تطوير الخوارزميات، وتصميم نماذج رياضية Mathematical Models لتوصيف مسألة ما، حيث تحتوي لغة البرمجة MATLAB على عدد كبير من التوابع الرياضية التي تدعم مجالات الهندسة المختلفة. كما تؤمن لنا هذه البرمجية السرعة العالية في تنفيذ التوابع الرياضية المستخدمة في الحسابات الخاصة بالأشعة والمصفوفات، تعود هذه السرعة العالية في التنفيذ بسبب استخدام مكاتب جرى ضمنها مسبقاً بناء التوابع بطريقة أمثلية. نستعرض فيما يلي أهم الطرائق الحسابية:

- الاستيفاء والطرائق العودية Interpolation and Regression.
- التفاضل والتكامل Differentiation and integration.
- جملة معادلات خطية Linear systems of equations.
- تحليل فوريه Fourier analysis
- القيم الذاتية والقيم المنفردة Eigenvalues and singular values
- معادلات تفاضلية Differential Equations

تؤمن لنا الطرائق الحسابية السابقة إجراء العديد من الحسابات الرياضيّة في مجالات مختلفة نذكر منها:

- Dealing with Matrices and Arrays التعامل مع المصفوفات والأشعة
- 2-D and 3-D Plotting and graphics رسومات وإظهارات بيانية ثنائية وثلاثية الأبعاد
- Linear Algebra الجبر الخطي
- Algebraic Equations المعادلات الجبرية
- Non-linear Functions التوابع غير الخطية
- Statistics الإحصاء
- Data Analysis تحليل المعطيات
- Differential Equations المعادلات التفاضلية
- Numerical Calculations الحسابات العدّدية
- Integration and differentiation التكاملات والاشتقاق
- Transforms such as Fourier and Laplace التحويلات مثل تحويل لابلاس وتحويل فوريه
- Curve Fitting

يتيح MATLAB أيضاً الحصول على منتجات إضافية متخصصة في عدة مجالات كالإحصاء، الأمثلة Optimization، ومعالجة الإشارة Signal Proccessing.

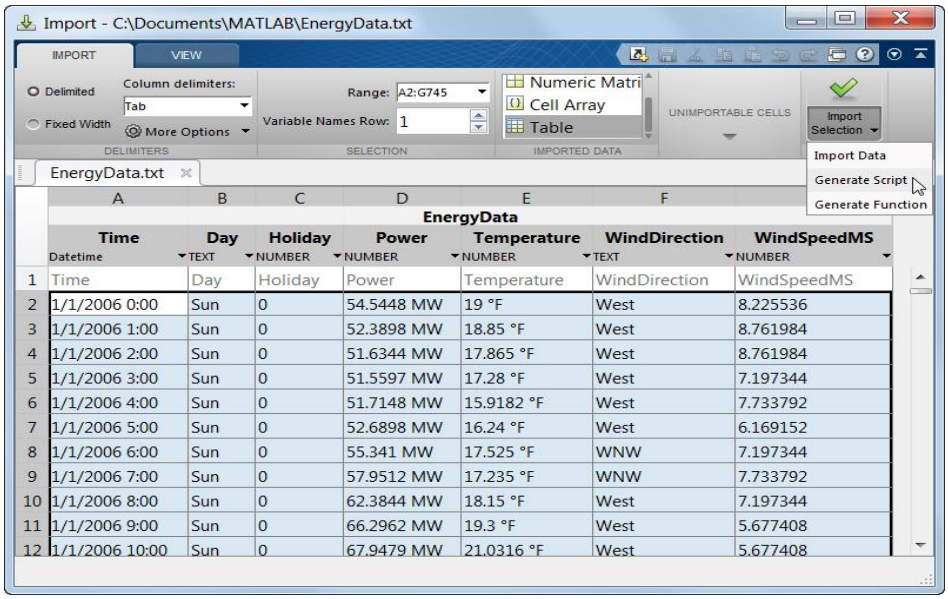


صورة على استخراج شكل منحني من مجموعة من المعطيات باستخدام Cubic Interpolation

3. تحليل المعطيات وإظهارها Data Analysis and Visualization

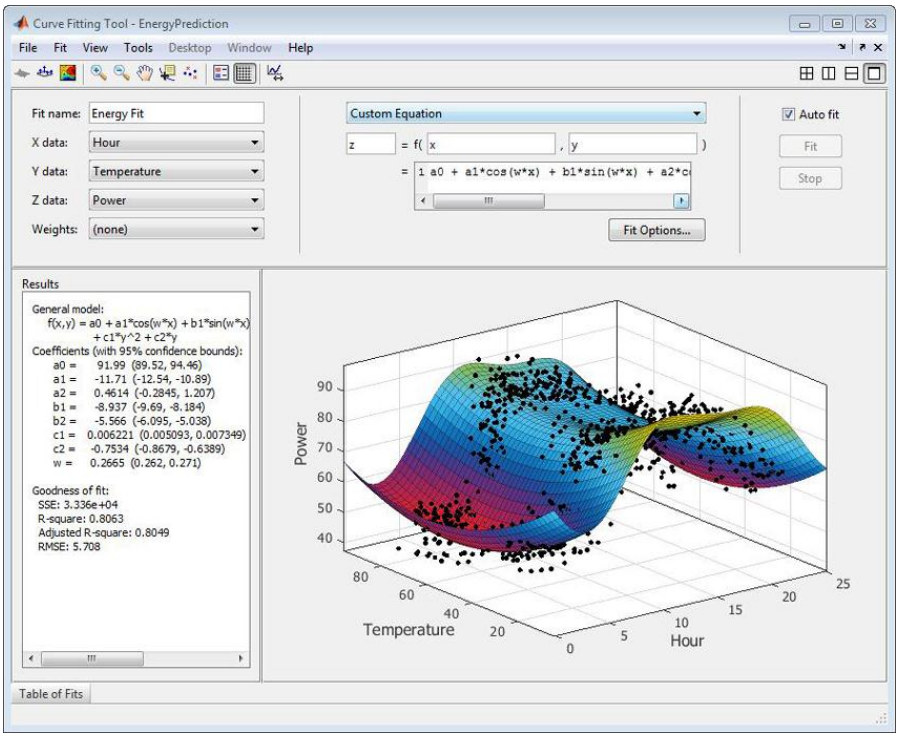
يؤمن لنا MATLAB عدّة أدوات Tools للحصول على، تحليل، وإظهار المعطيات وأخيراً يمكن الحصول على ملفات document تحتوي على النتائج الخاصة بالمستخدم كأشكال ومنحنيات ورسومات ونتائج عدّدية.

تتيح لنا البرمجية، من أجل الحصول على المعلومات، إمكانية النفاذ إلى معطيات موجودة ضمن ملفات text، قواعد معطيات Databases، Microsoft Excel، صور، صوت وملفات فيديو وذلك على اختلاف الصيغ للمعطيات حيث جرى بناء عدد كبير من توابع دخل/خرج I/O وهي اختصار ل Input/Output. كما يمكن الحصول على المعطيات من الحاسب عن طريق البوابة التسلسلية Serial Port أو كرت الصوت Sound Card.



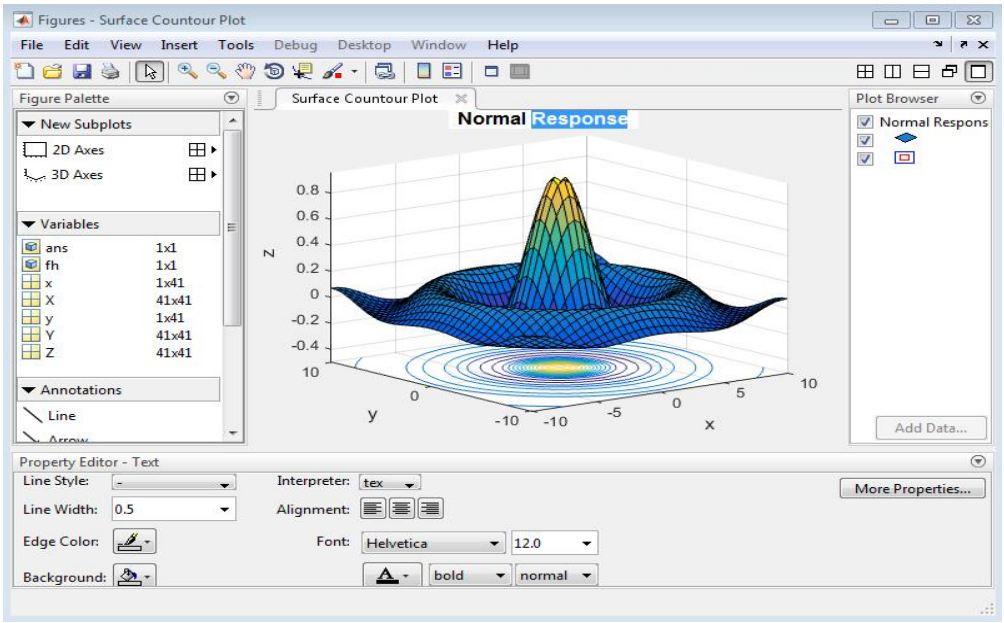
مثال على جلب import معطيات إلى MATLAB باستخدام Import Tool حيث يقوم MATLAB بشكل اتوموماتيكي بإنشاء script او تابع يحوي على المعطيات التي تم جلبها بواسطة الأداة.

من أجل تحليل المعطيات يوجد العديد من الأدوات التي تتيح لنا عملية التحليل، نذكر منها Curve Fitting Tool المبينة في الشكل حيث يتم إيجاد صيغة رياضية تحليلة لسطح يتلائم مع المعطيات التجريبية.



Fitting a surface to data with a custom model using MATLAB and Curve Fitting Toolbox.

من أجل إظهار النتائج إضافةً لما سبق يوجد أدوات وتوابع بمختلف الأحداثيات كالقطبية والديكارتية والكروية و منحنيات ثنائية الأبعاد أو ثلاثية الأبعاد. الشكل التالي يبين لنا الخيارات العديدة الممكن إضافتها على المنحنيات الناتجة بهدف عرضها بالطريقة، والألوان والحجم المناسب إضافةً لإمكانية إدراج تسميات للمحاور والأشكال.



موارد لتعلم MATLAB

يوجد العديد من الموارد المفيدة والمتوفرة من قبل MathWorks® (موقع الشركة هو https://www.mathworks.com) يرجى ذكر موقع الشركة ونصح الطلاب بزيارته والتي تمكننا من اكتشاف المزيد والتعلم عن MATLAB، هذه الموارد هي:

1. الوثائق (Documentation)
2. التوابع والرمازات على شكل أمثلة تعليمية (Functions and Code Examples)
3. الفيديوهات والوثائق التعليمية (Videos and Tutorials)

تؤمن الموارد السابقة شرح التعليمات والتوابع، والتغييرات التي طرأت على البرمجية، تبعاً للإصدارات المختلفة (Release)، من حيث:

1- التعليمات 2- الأدوات Toolboxes الإضافية 3- التوابع حديثة التنجيز في كافة المجالات.

ننوه هنا إلى أمر مهم وهو إن لم تكن على دراية كافية بمطلبات المقرر prerequisities فهذه ليست بمشكلة كبيرة حيث تعتبر برمجية MATLAB نفسها إحدى أفضل المصادر لتعلم هذه المفاهيم بسرعة وفاعلية كبيرة فهي تمتاز بـ"مساعدة HELP" متميّزة واستثنائية تقوم بشرح التعليمات وطريقة استخدامها بشكل واضح وصحيح ومفصّل إضافةً للمفاهيم المرتبطة بالتعليمة، والعلاقات الرياضية في حال حاجتها، وتعتبر من أفضل المراجع التي ينصح باللجوء إليها قبل استخدام أي تابع لضمان الاستخدام الصحيح وتلافي الأخطاء. حيث يتم كتابة help ثم اسم التعليمة

من أجل تخصيص البحث والحصول على مساعدة مدعومة بأمثلة حول تعليمة يمكن استخدام أيضاً doc ثم اسم التعليمة

كما أن البحث ضمن الوب وإيجاد العديد من الملفات التعليمية (Tutorials) قد يكون مفيداً عند اختيار المادة المناسبة والمصدر الموثوق، إضافةً لوجود عدد كبير من الكتب المتعلقة بتعليم MATLAB و Simulink و Toolboxes.

بعض المواقع المفيدة

http://www.mathworks.com/academia/student_center/tutorials/ للحصول على Tutorials

http://www.mathworks.com/products/MATLAB/videos.html للحصول على فيديوهات تعليمية

http://www.mathworks.com/company/events/webinars/index.html?id=&language=en فيديوهات لبعض الندوات على الانترنت

http://www.mathworks.com/MATLABcentral/ وهو موقع لتبادل الملفات والخبرات MATLA Central File Exchange حيث يوجد العديد من الأمثلة التعليمية وبعض الإجابات عن مشاكل واجهت المستخدمين وقد

تم تقديم حلول مفيدة لها وعدد كبير من التوابع المفيدة والمنجزة من قبل مستخدمي MATLAB وقد وضعت ضمن الموقع للاستفادة منها.

كما يمكن إنشاء حساب Account في موقع الشركة وهذا أمر مهم للغاية.

إصدارات البرمجية

نعرض في هذه الفقرة بعض إصدارات البرمجية، حيث يتم في كل فترة زمنية تحديث البرمجية من حيث الأداء، التوافق مع البرمجيات الأخرى compatibility، الشكل والتفاعل مع المستخدم، تصحيح الأخطاء في حال وجودها في الإصدارات السابقة، إضافة توابع في مجالات مختلفة تسهل عمل المستخدم، إضافة إمكانيات جديدة في إظهار الرسوم والبيانات واستيراد وتصدير المعطيات وإنشاء الواجهات البيانية، وفي بعض الأحيان يتم إضافة Toolbox في مجال جديد لم يكن موجوداً في الإصدارات السابقة. بعض الإصدارات ل MATLAB

Previous Releases

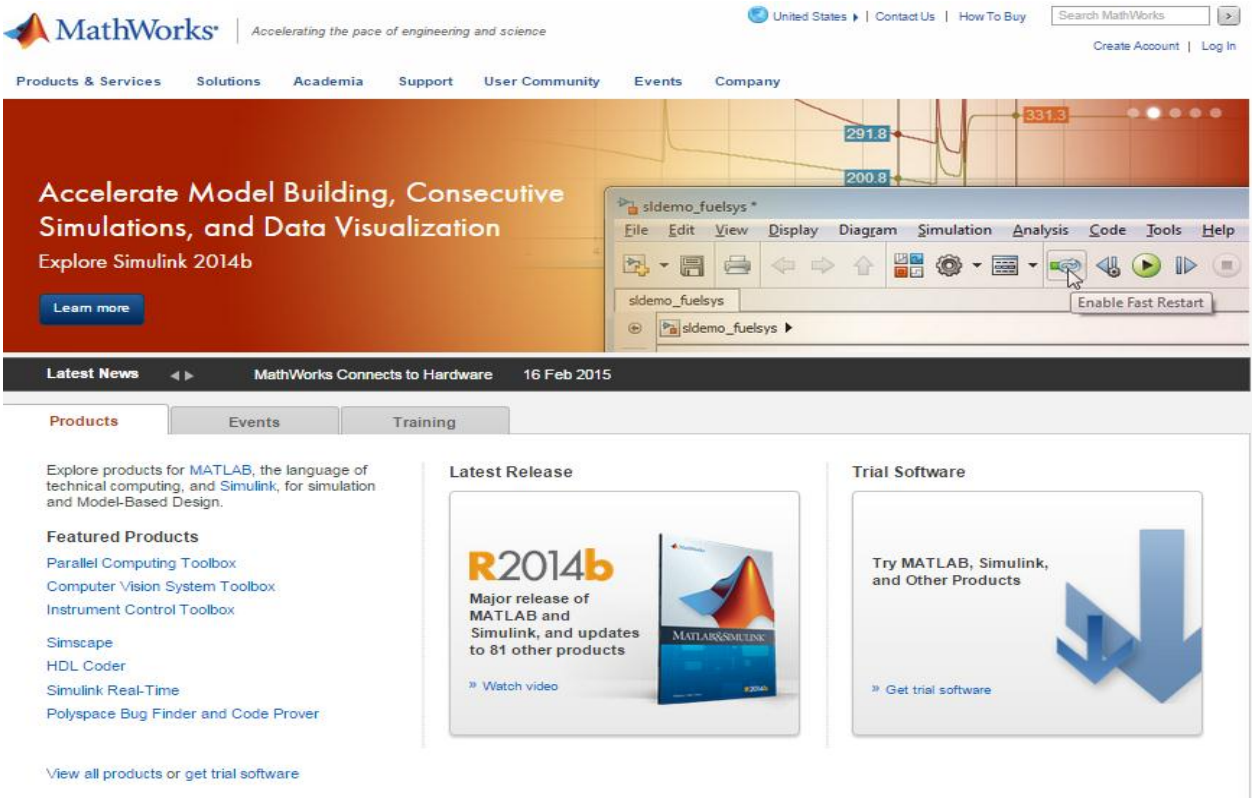
- **R2014b** (Version 8.4) - 2 Oct 2014
- **R2014a** (Version 8.3) - 6 Mar 2014
- **R2013b** (Version 8.2) - 5 Sep 2013
- **R2013a** (Version 8.1) - 7 Mar 2013
- **R2012b** (Version 8.0) - 11 Sep 2012
- **R2012a** (Version 7.14) - 1 Mar 2012
- **R2011b** (Version 7.13) - 1 Sep 2011
- **R2011a** (Version 7.12) - 8 Apr 2011
- **R2010b** (Version 7.11) - 3 Sep 2010
- **R2010a** (Version 7.10) - 5 Mar 2010
- **R2009b** (Version 7.9) - 4 Sep 2009
- **R2009a** (Version 7.8) - 6 Mar 2009
- **R2008b** (Version 7.7) - 9 Oct 2008
- **R2008a** (Version 7.6) - 1 Mar 2008
- **R2007b** (Version 7.5) - 1 Sep 2007
- **R2007a** (Version 7.4) - 1 Mar 2007

ويتم العمل ضمن شركة MathWorks® على إصدار R2015b، يمكن عن طريق الرابط التالي
http://www.mathworks.com/help/MATLAB/release-notes.html

الاطلاع على ملاحظات حول الإصدارات المختلفة وذلك للتعرف على ما هو جديد من توابع ومكاتب و toolboxes أو ما تم إصلاحه من أخطاء في الإصدارات الجديدة.

تنزيل برمجية MATLAB

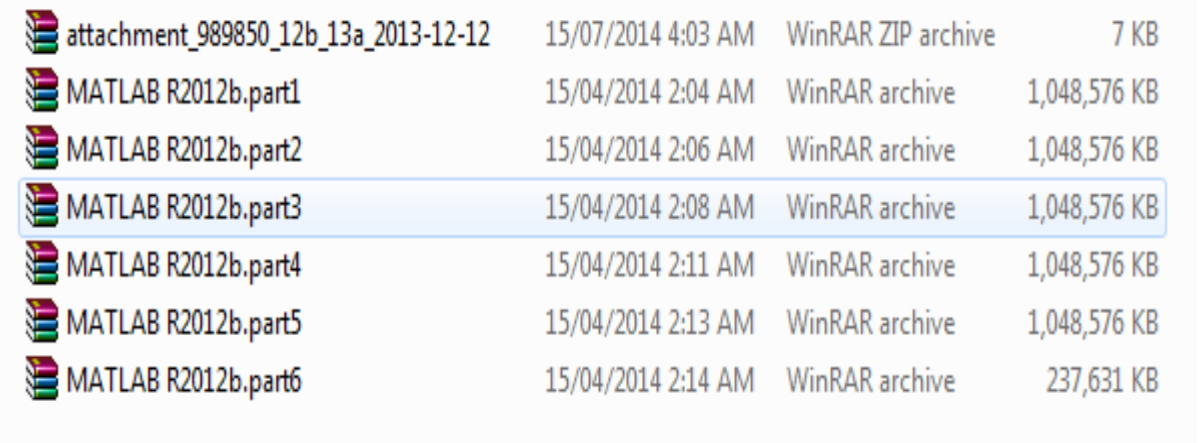
تستطيع تحميل برمجية MATLAB من خلال الدخول إلى موقع شركة MathWorks® الالكتروني وهو http://www.mathworks.com وهذه الصورة هي واجهة الموقع



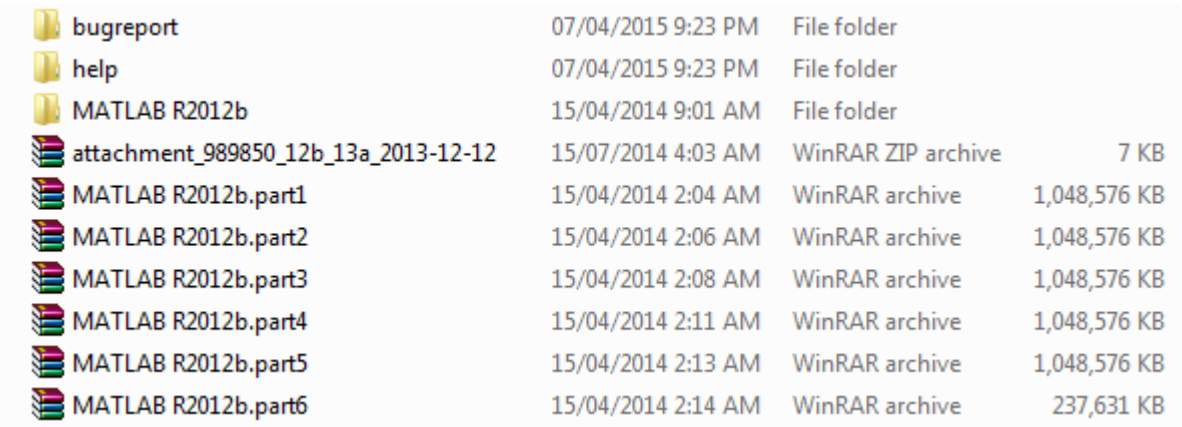
إن هذه البرمجية غير مجانية، يمكن من الموقع الحصول على نسخة تجريبية Trial تعمل لعدّة أيام فقط، ويمكن الاطلاع على مزايا البرمجية والأدوات الممكن إضافتها للبرمجية والمتخصصة بمجالات مختلفة ضمن الموقع.

من أجل تنزيل البرمجية على الحاسب الخاص بك يرجى اتباع التعليمات التالية:

نفتح المجلد MATLAB R2012b يوجد عدد من الملفات المضغوطة بلاحقة rar. عددها 6 هي الملفات MATLAB R2012b.part1 ..... MATLAB R2012b.part6 إضافةً لملف آخر مضغوط اسمه attachment

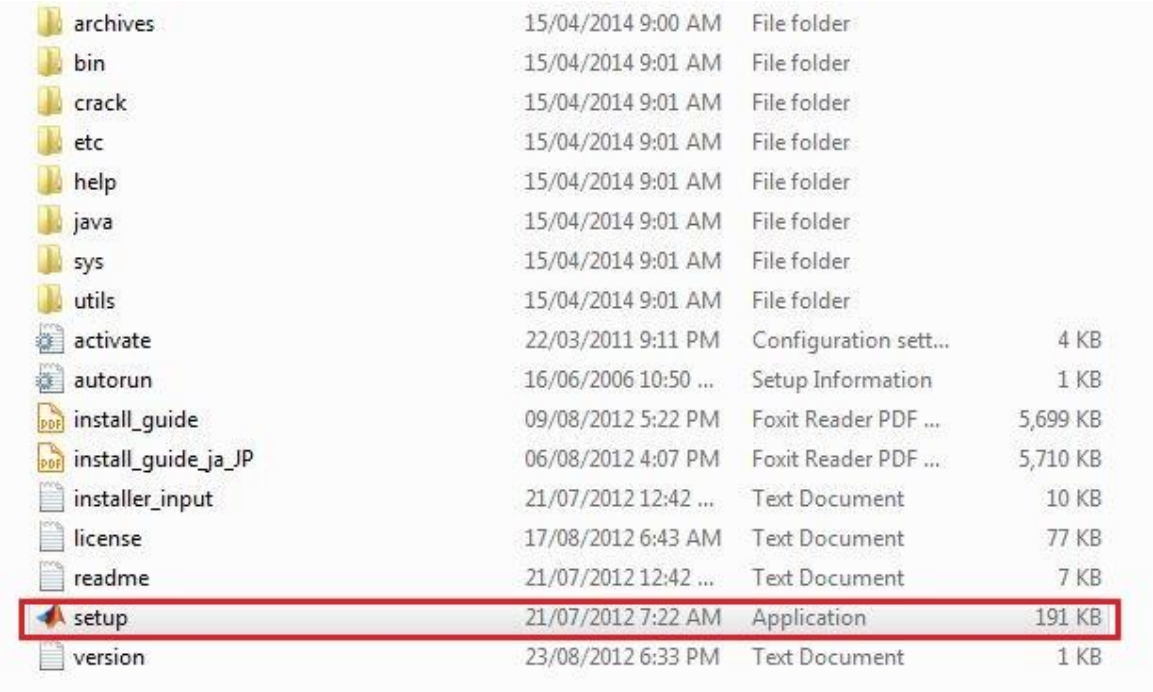


نقوم بفك ضغط كل منها الملفات MATLAB R2012b.part1 إلى MATLAB R2012b.part6 ستعطي مجلد اسمه MATLAB R2012b أم عند فك ضغط الملف attachement سيتم الحصول على مجلدين هما help و bugreport



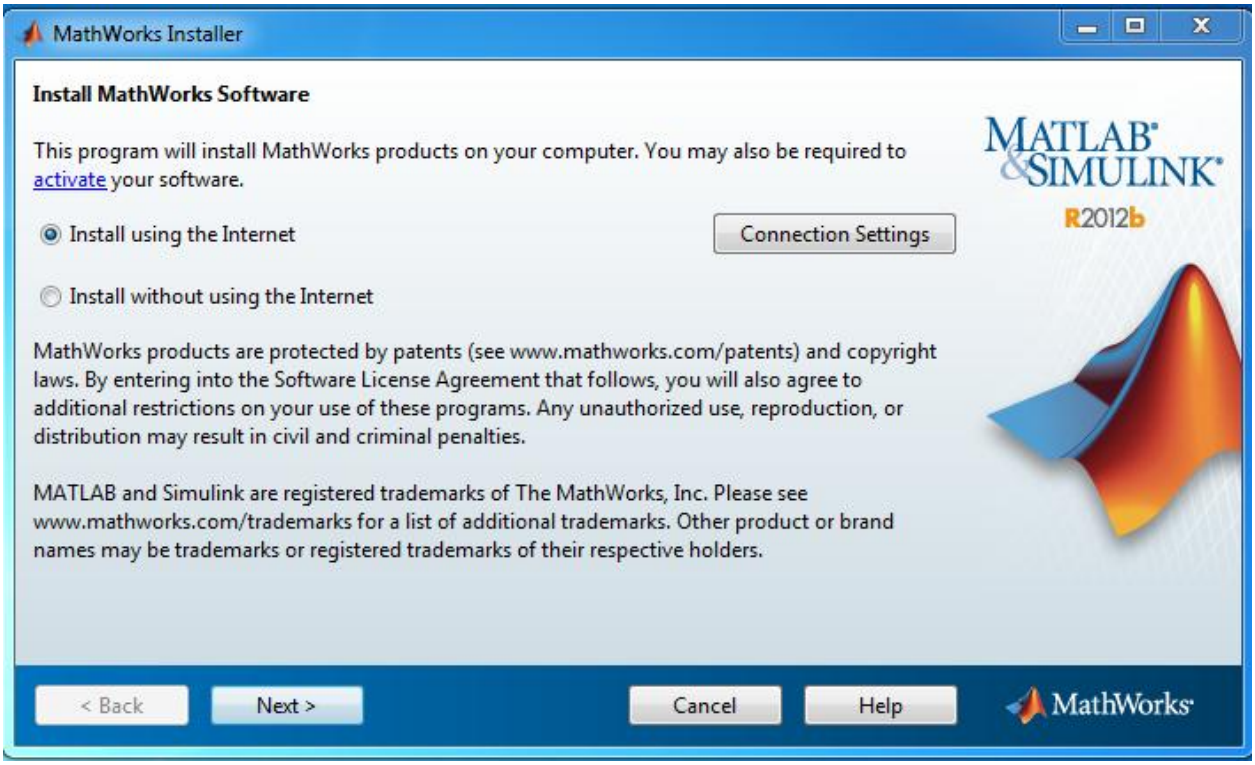
bugreport	07/04/2015 9:23 PM	File folder	
help	07/04/2015 9:23 PM	File folder	
MATLAB R2012b	15/04/2014 9:01 AM	File folder	
attachment_989850_12b_13a_2013-12-12	15/07/2014 4:03 AM	WinRAR ZIP archive	7 KB
MATLAB R2012b.part1	15/04/2014 2:04 AM	WinRAR archive	1,048,576 KB
MATLAB R2012b.part2	15/04/2014 2:06 AM	WinRAR archive	1,048,576 KB
MATLAB R2012b.part3	15/04/2014 2:08 AM	WinRAR archive	1,048,576 KB
MATLAB R2012b.part4	15/04/2014 2:11 AM	WinRAR archive	1,048,576 KB
MATLAB R2012b.part5	15/04/2014 2:13 AM	WinRAR archive	1,048,576 KB
MATLAB R2012b.part6	15/04/2014 2:14 AM	WinRAR archive	237,631 KB

نقوم بفتح المجلد MATLAB R2012b ونختار من ضمنه setup

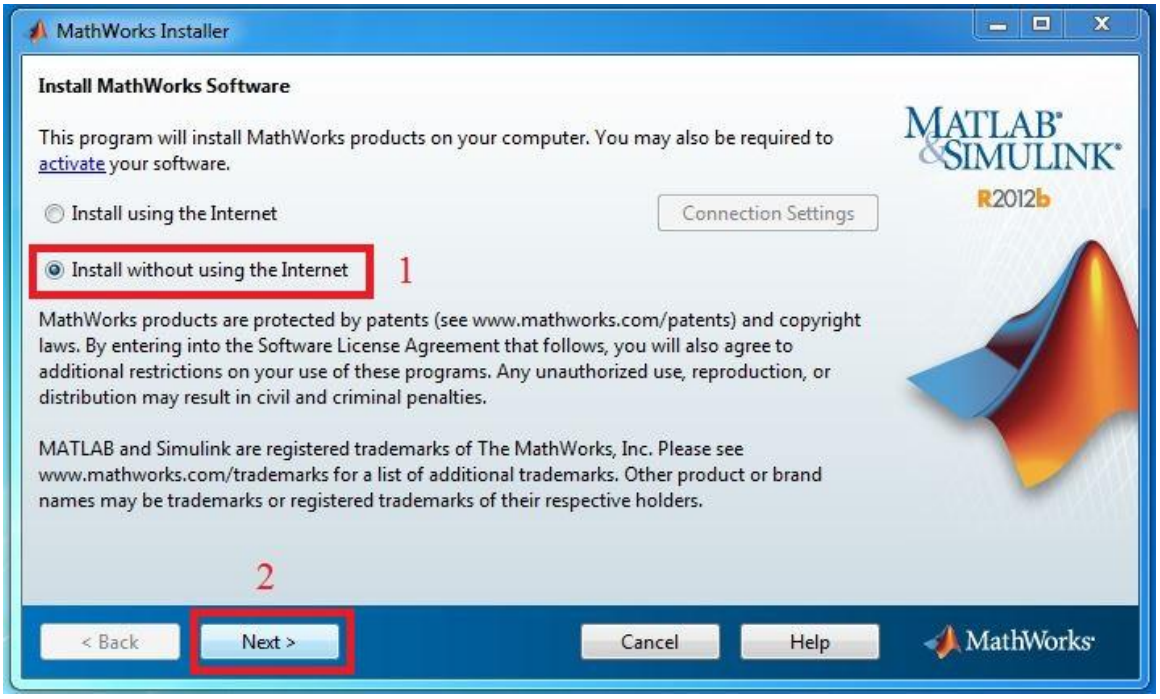


archives	15/04/2014 9:00 AM	File folder	
bin	15/04/2014 9:01 AM	File folder	
crack	15/04/2014 9:01 AM	File folder	
etc	15/04/2014 9:01 AM	File folder	
help	15/04/2014 9:01 AM	File folder	
java	15/04/2014 9:01 AM	File folder	
sys	15/04/2014 9:01 AM	File folder	
utils	15/04/2014 9:01 AM	File folder	
activate	22/03/2011 9:11 PM	Configuration sett...	4 KB
autorun	16/06/2006 10:50 ...	Setup Information	1 KB
install_guide	09/08/2012 5:22 PM	Foxit Reader PDF ...	5,699 KB
install_guide_ja_JP	06/08/2012 4:07 PM	Foxit Reader PDF ...	5,710 KB
installer_input	21/07/2012 12:42 ...	Text Document	10 KB
license	17/08/2012 6:43 AM	Text Document	77 KB
readme	21/07/2012 12:42 ...	Text Document	7 KB
setup	21/07/2012 7:22 AM	Application	191 KB
version	23/08/2012 6:33 PM	Text Document	1 KB

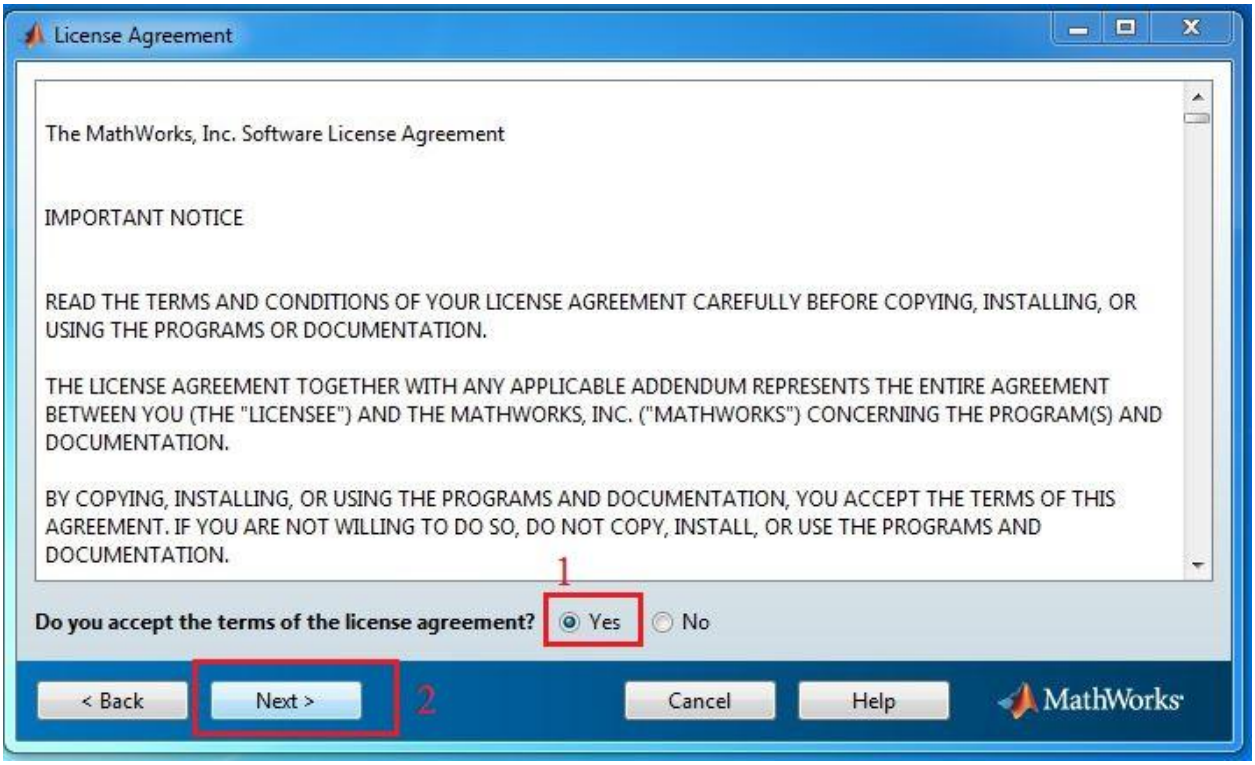
سيتم فتح نافذة على الشكل التالي



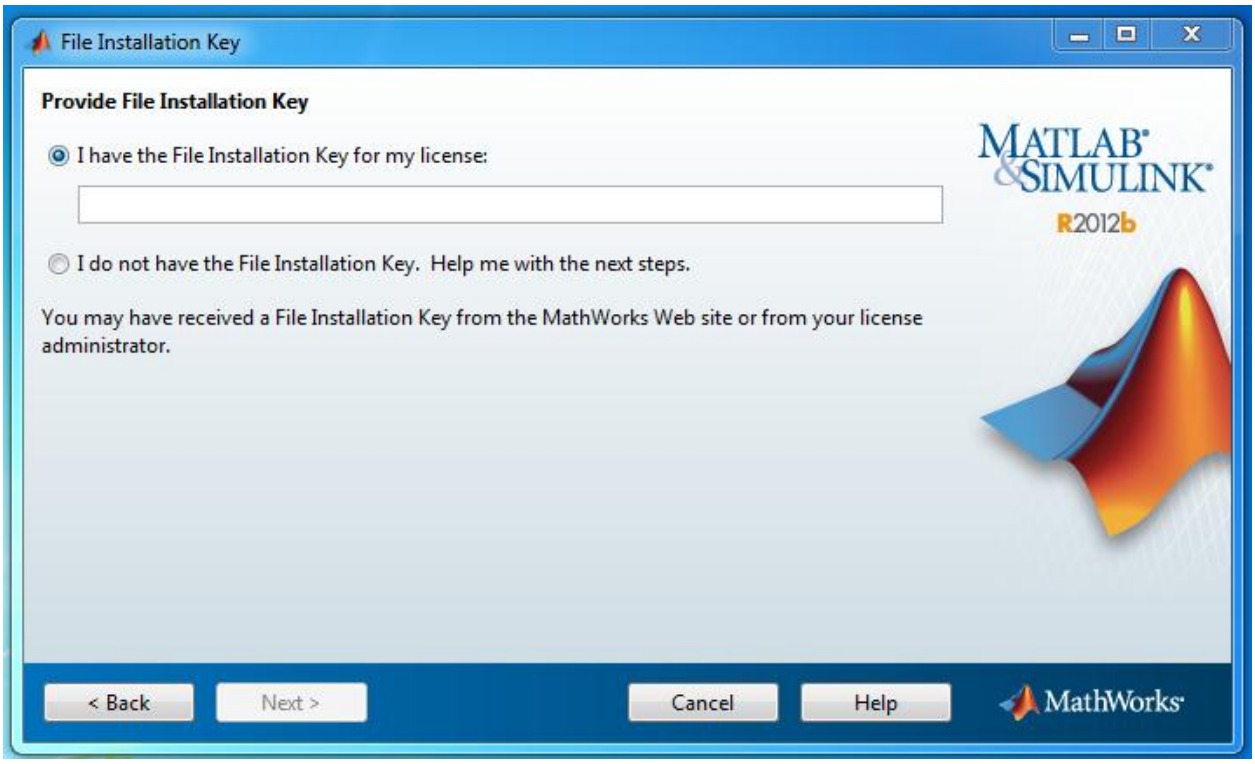
اختر Install without using the Internet ثم اختر next كما في الشكل



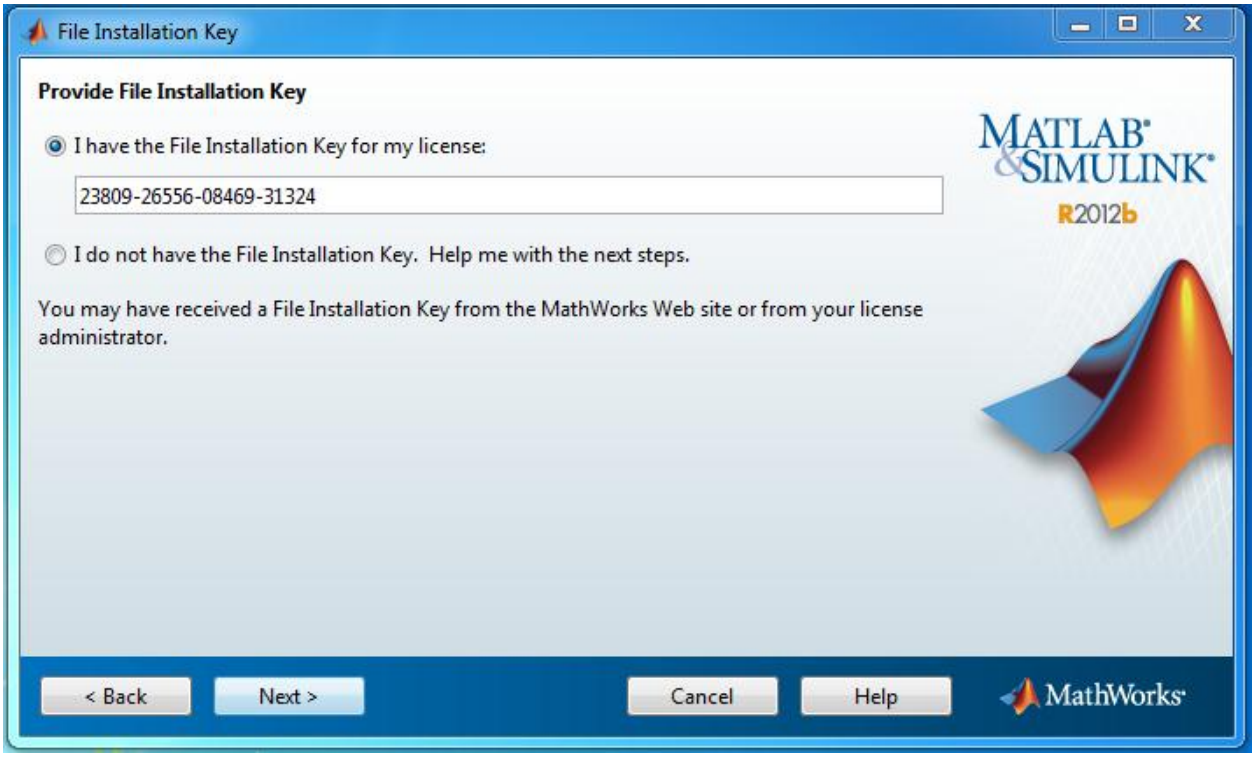
ستظهر واجهة تحتوي على Do you accept the terms of the license agreement? اختر yes ثم اضغط next.



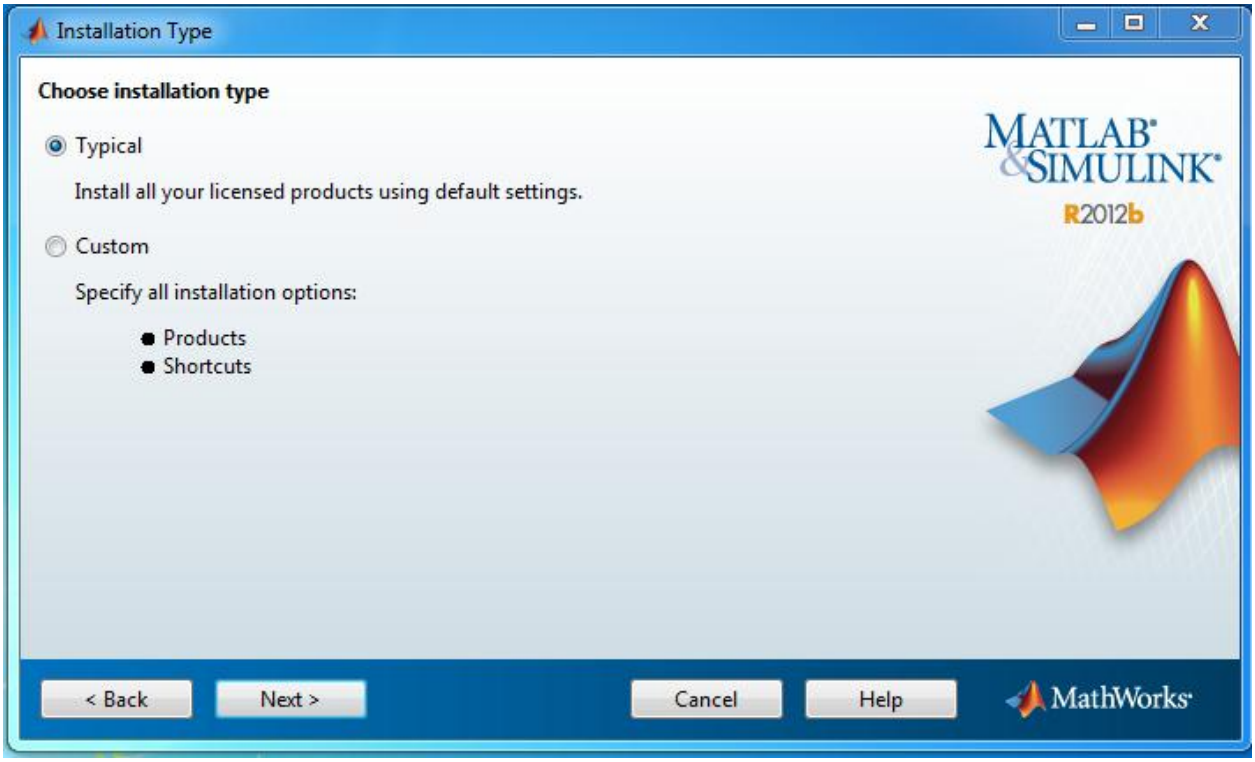
ستظهر أمامك الواجهة التالية، اختر I have the File Installation Key for my license



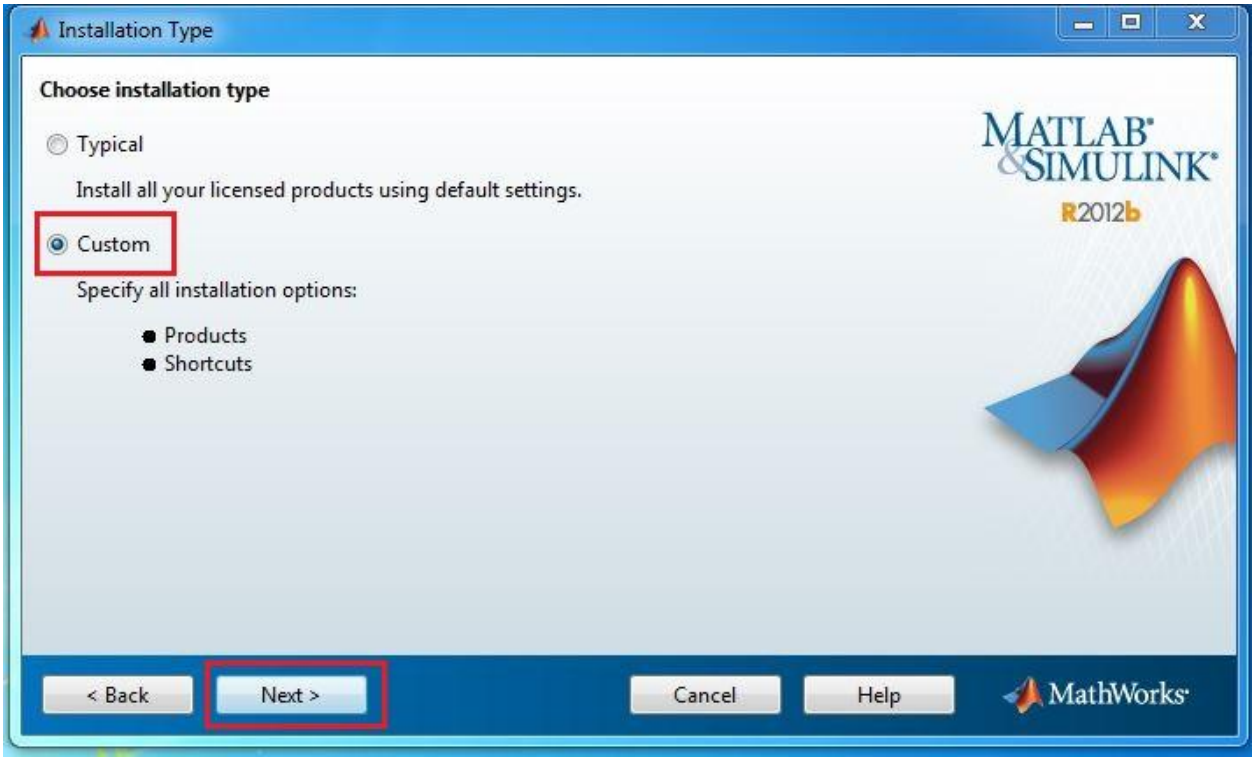
ثم ضع ضمن الحقل الرقم التسلسلي التالي: 23809-26556-08469-31324 كما في الشكل واضغط على next



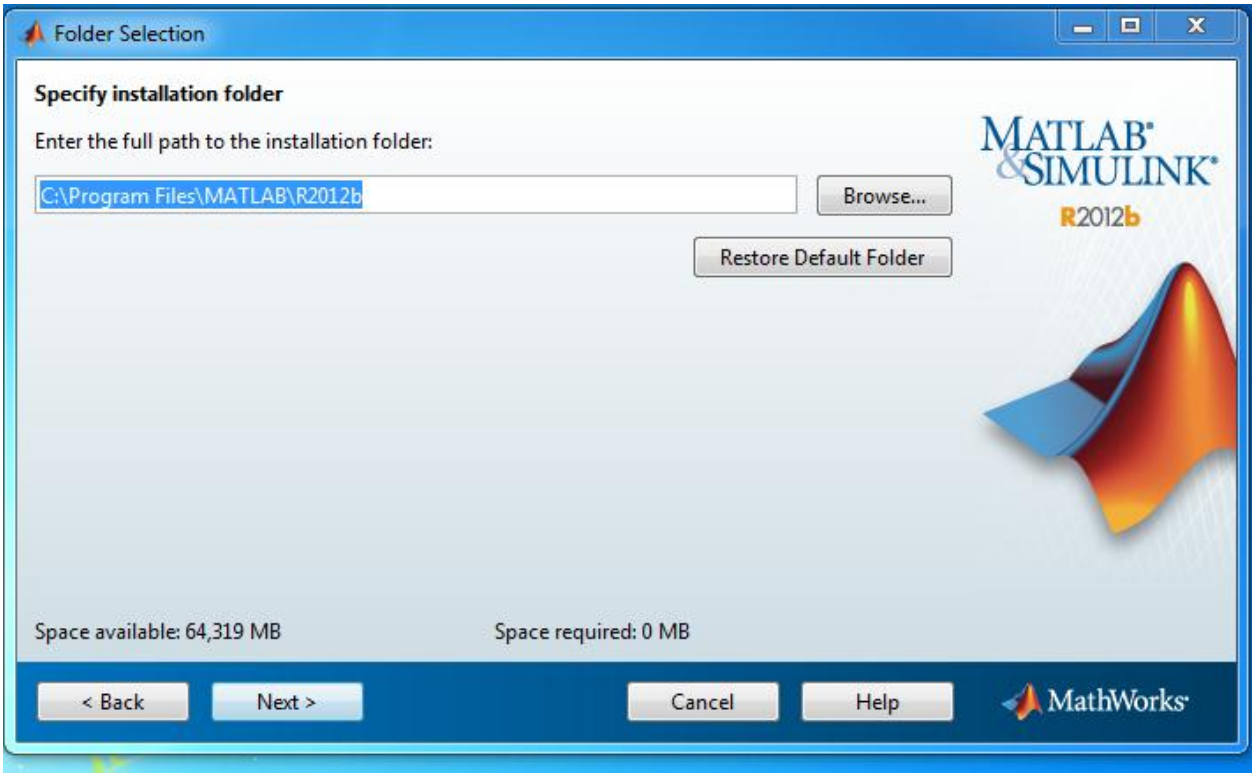
ستظهر أمامك الواجهة التالية



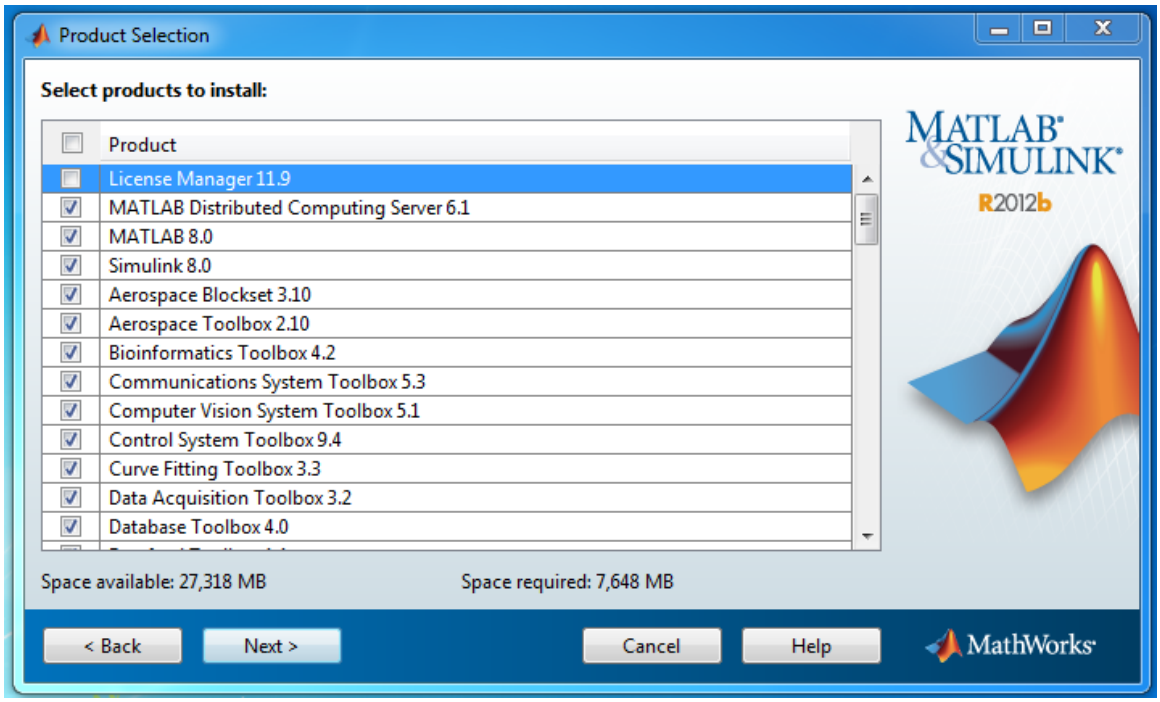
قم باختيار Custom ثم اضغط على next



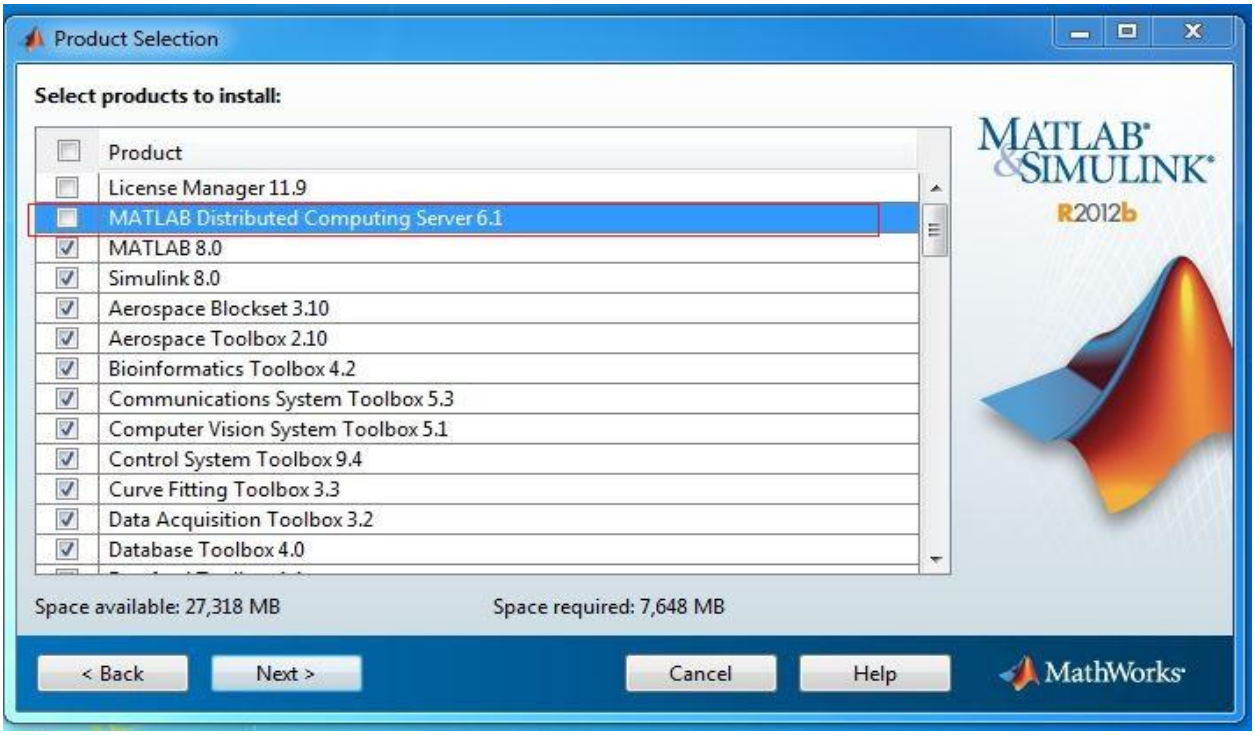
ستظهر أمامك الواجهة التالية وهذه الخطوة هي خطوة تحديد مجلد التنزيل



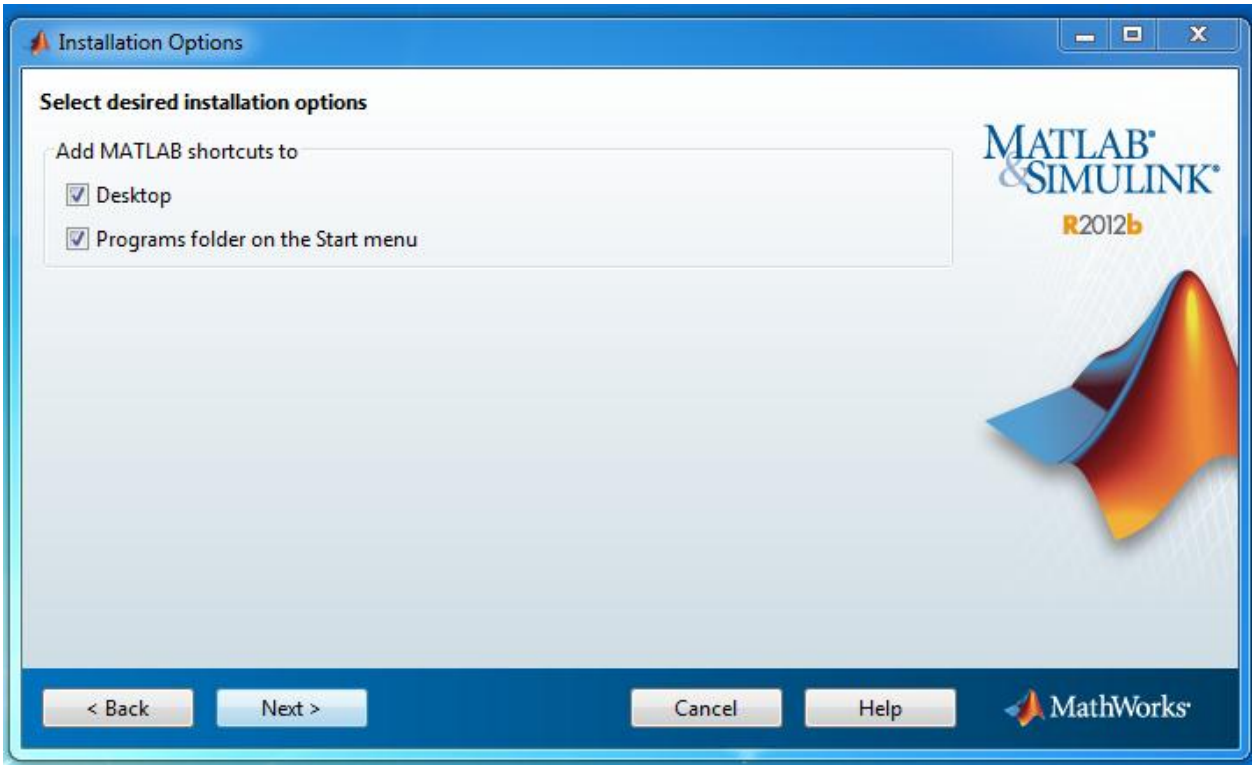
الواجهة تطلب منك تحديد المسار الكامل لتنزيل الرمجية ضمن مجلد التنزيل، يفضل استخدام القرص C لكن الأمر تابع للشخص، يمكن تغيير المسار عن طريق Browse. ثم اضغط next ستظهر أمامك الواجهة التالية



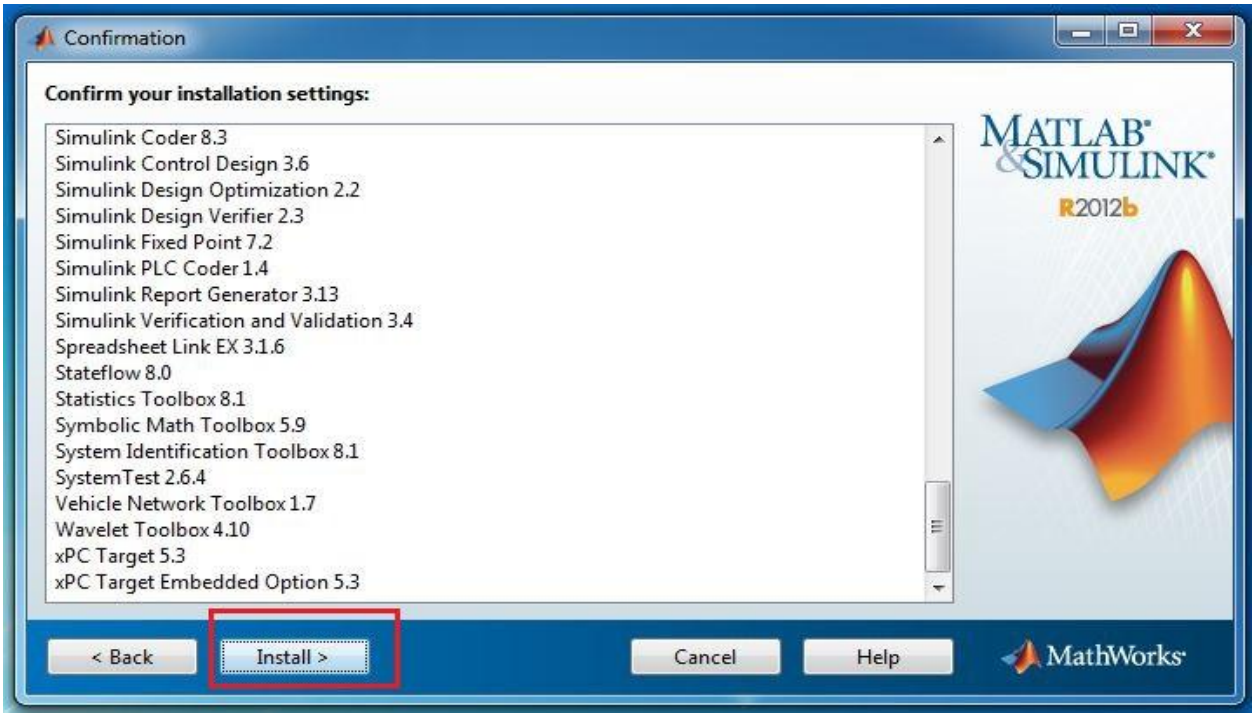
قم بإلغاء اختيار MATLAB Distributed Computing Server 6.1 كما يوضح الشكل



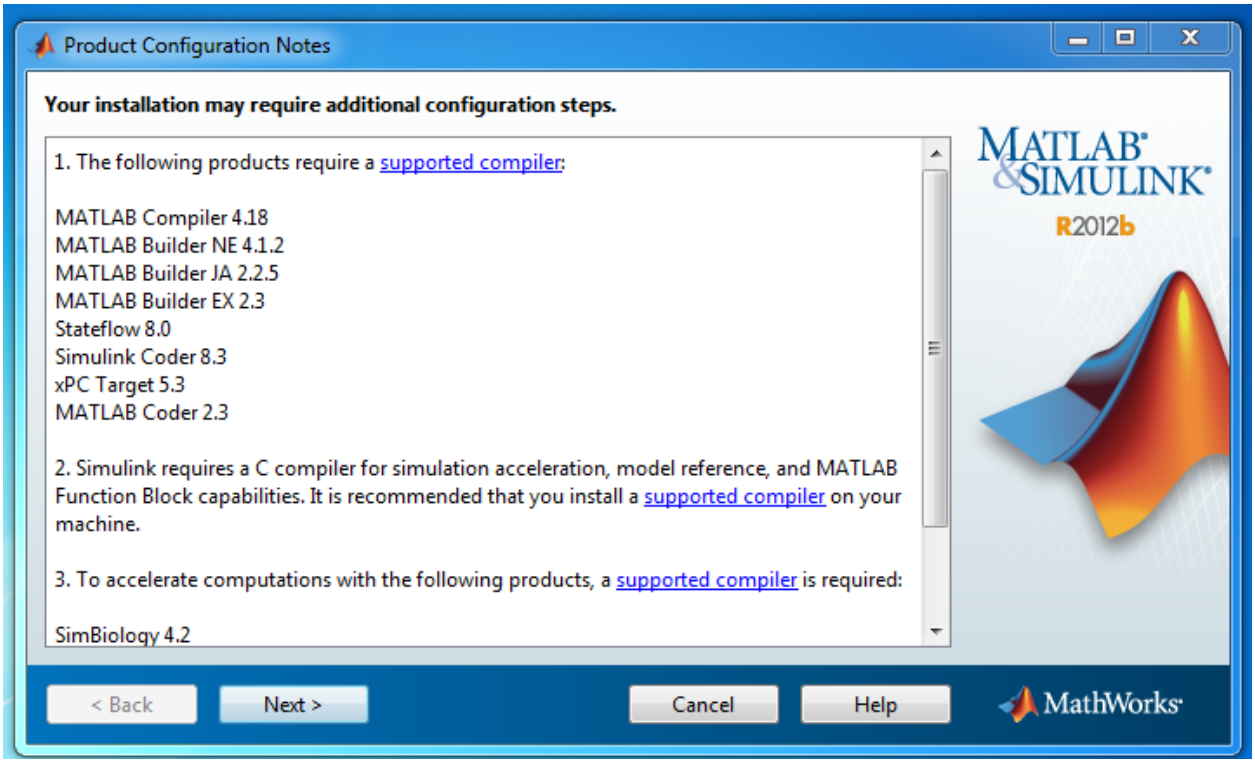
ثم اضغط على next ستظهر أمامك واجهة تطلب منك أن تختار في وضع اختصارات shortcuts خاصة بالبرمجية يمكن أم تختار أن تضع اختصار ضمن شطح المكتب desktop و كذلك الأمر في Start Menu كما يوضح الشكل



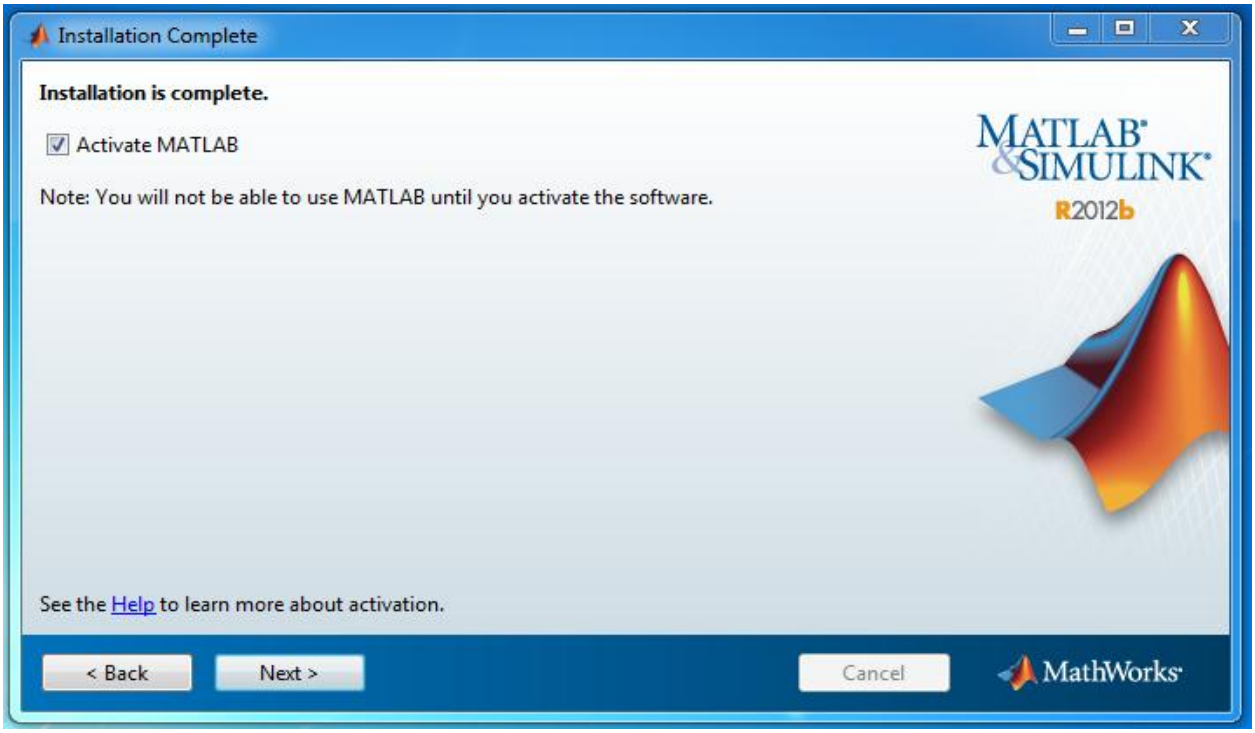
ثم ستظهر أمامك الواجهة التالية التي تطلب منك إعطاء أمر التنزيل، اختر Install لبدء تنزيل البرمجية



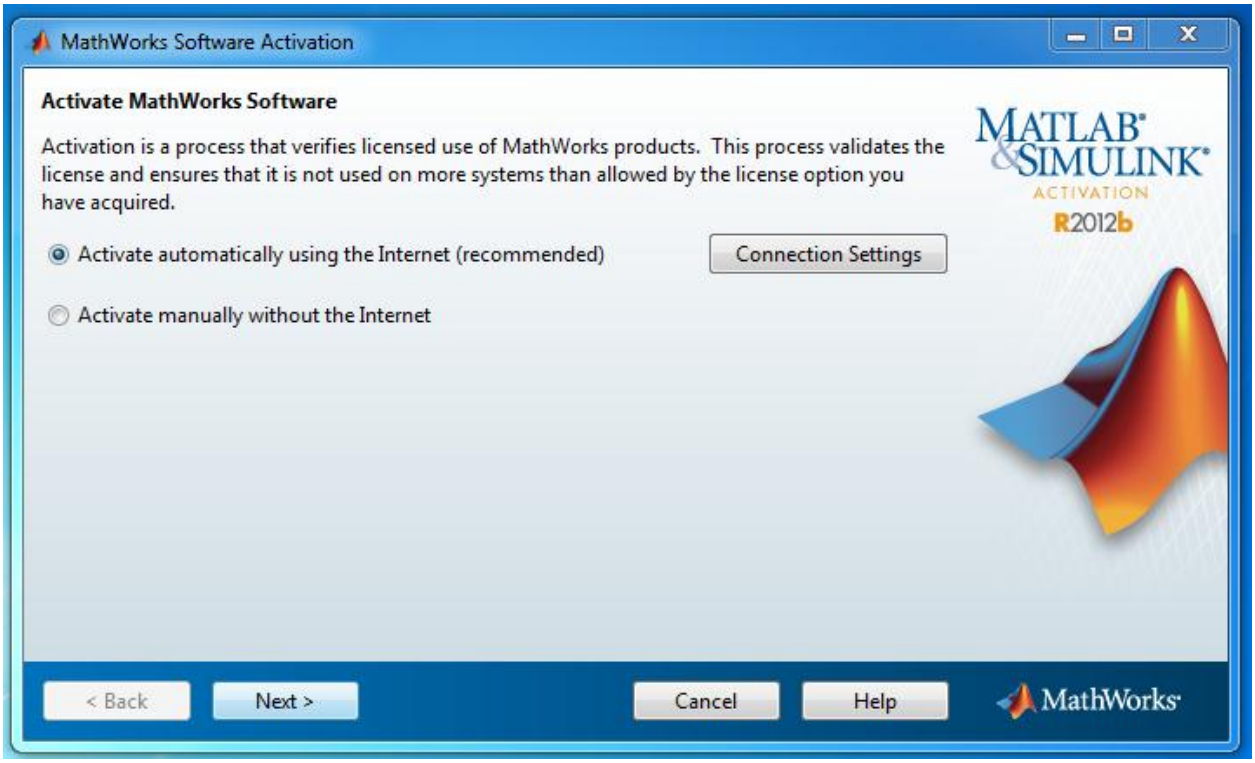
بعد الانتهاء من التنزيل، حيث يجب التنويه إلى أن التنزيل يتطلب وقت كبير حوالي ربع أو نصف ساعة، ستظهر أمامك الواجهة التالية، اختر next



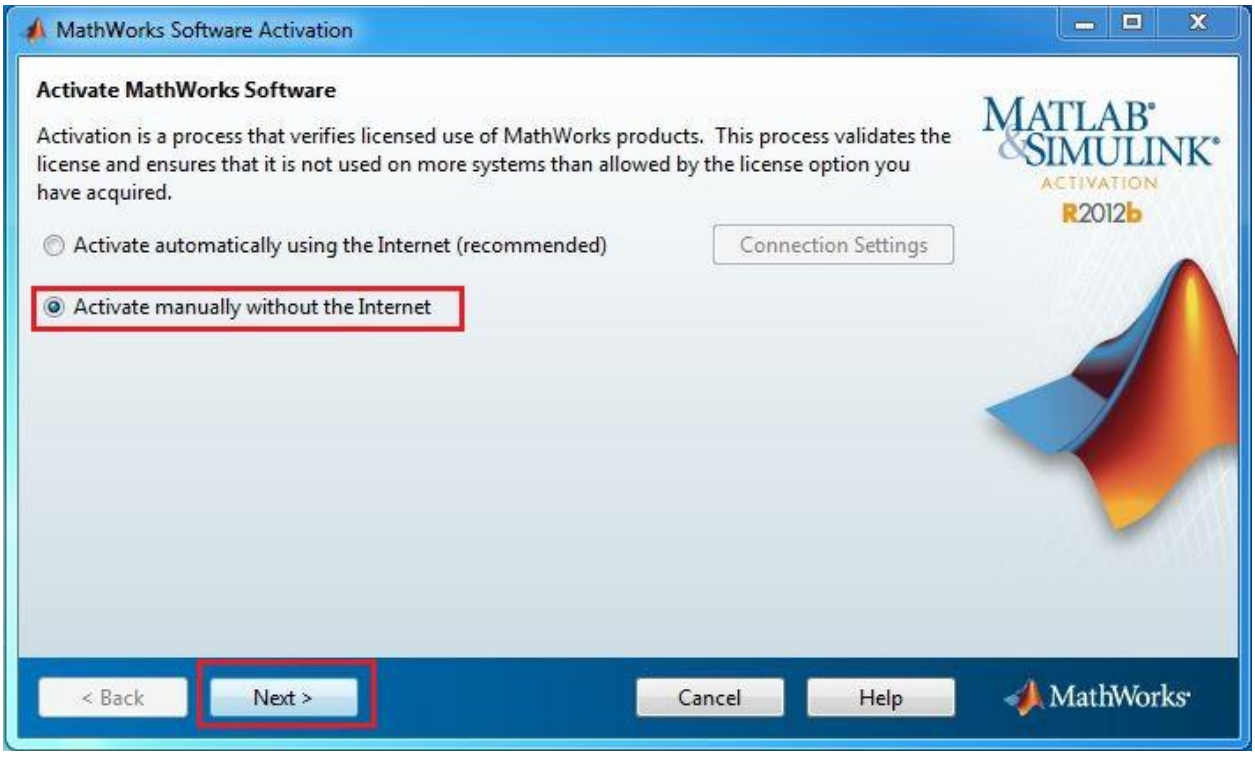
ثم ستظهر الواجهة التي تطلب منك تفعيل البرمجية، في حالة عدم التفعيل لن تكون قادراً على استخدام البرمجية، اختر next



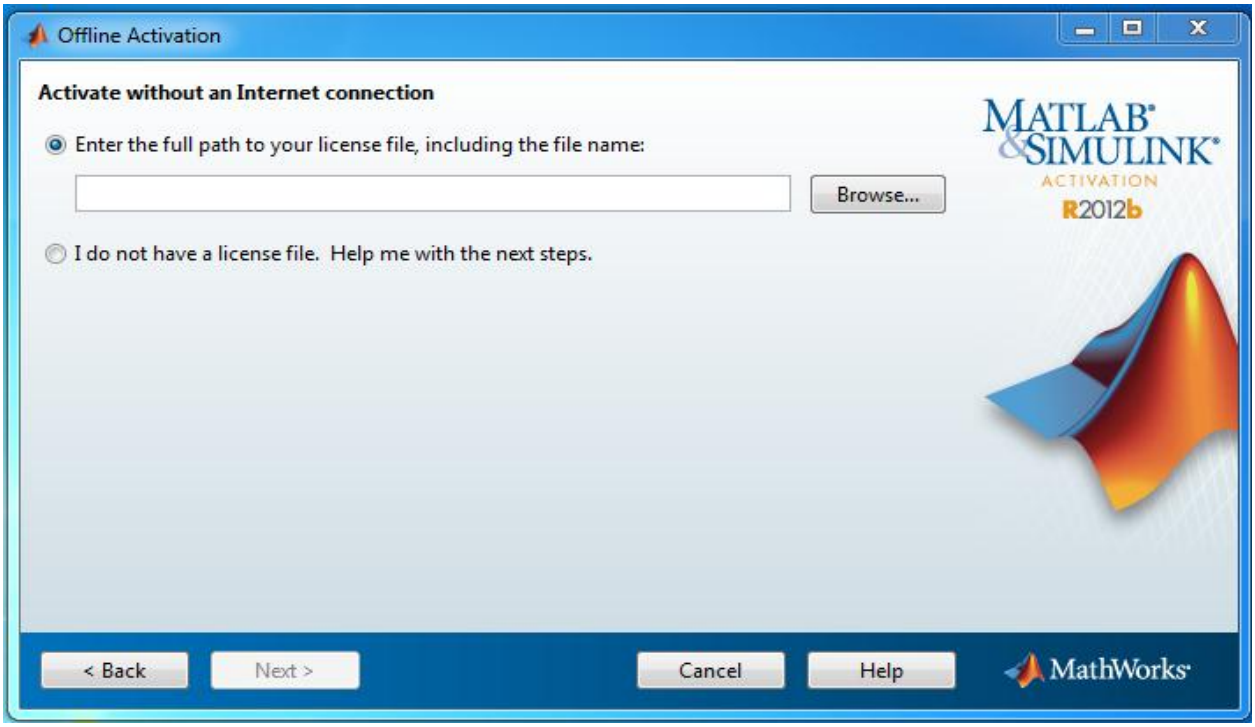
ستظهر أمامك الواجهة التالية



اختر Activate manually without using the Internet ثم اضغط على next

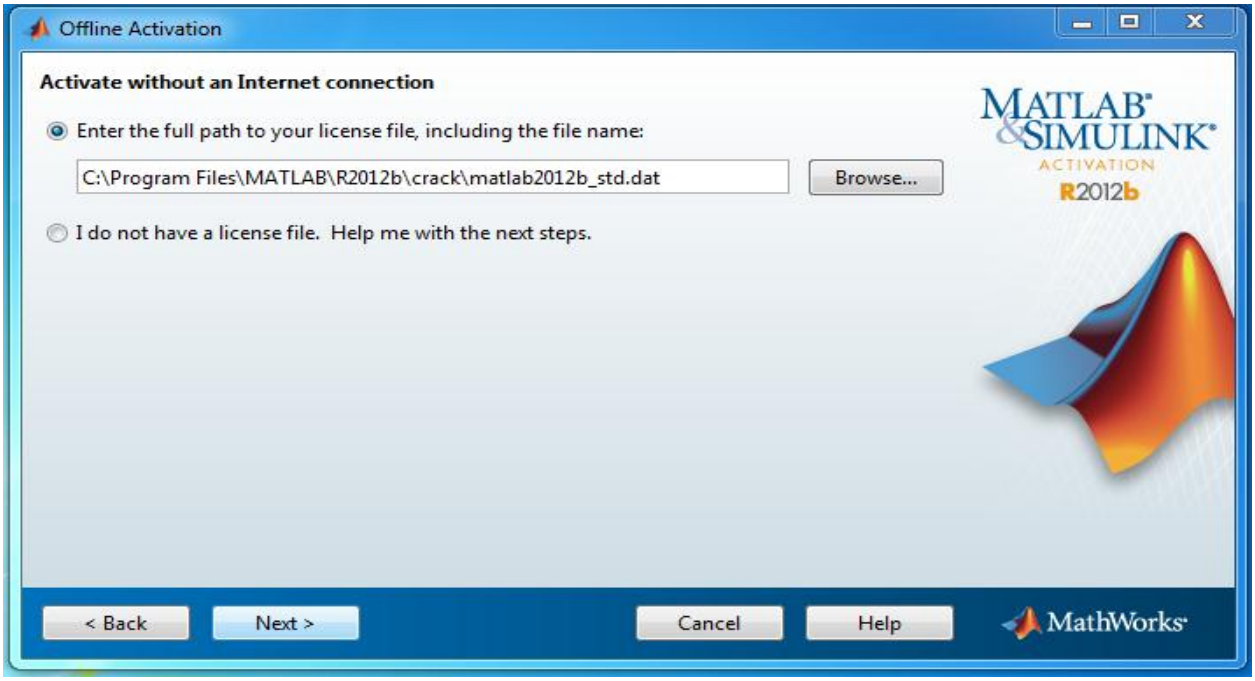


ستظهر أمامك الواجهة التالية قم باختيار Enter the full path to your license file, including the file name

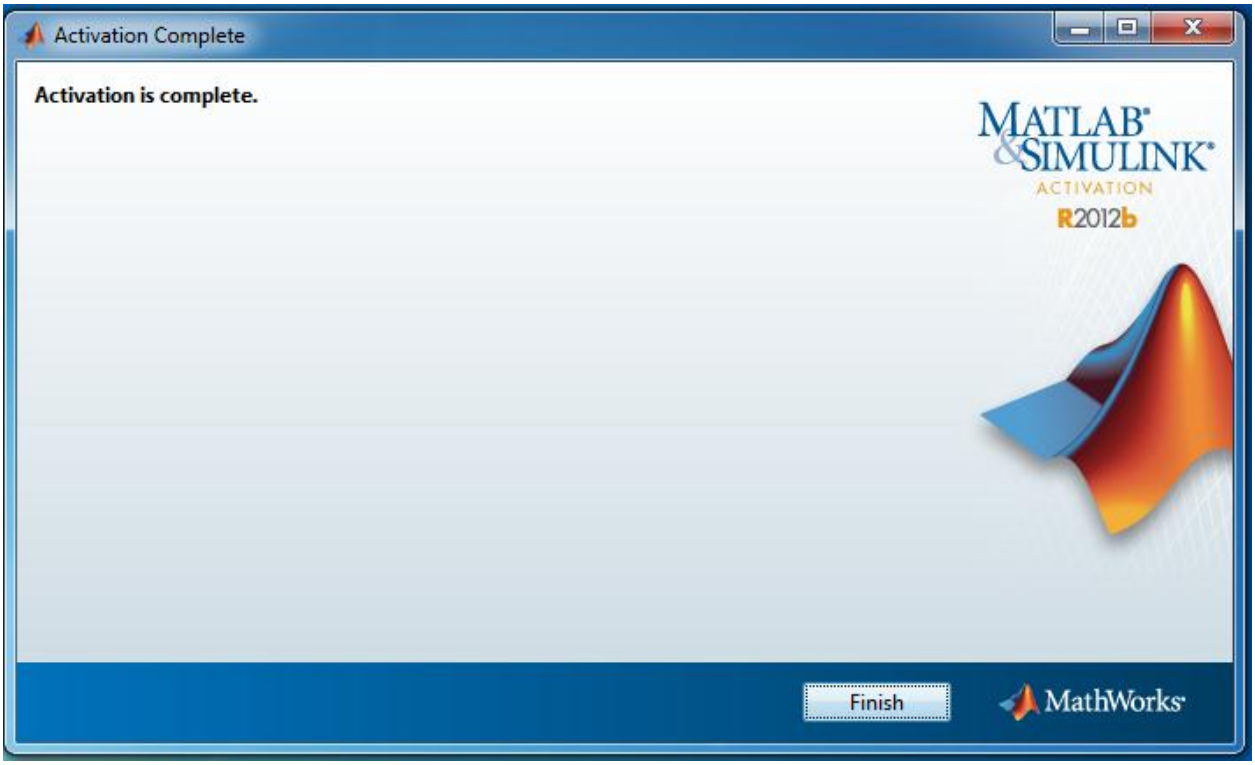


يوجد لتفعيل البرمجية الملف 'MATLAB2012b_std.dat' الموجود ضمن المجلد crack ضمن المجلد MATLAB R2012b

يفضل نسخ مجلد crack كاملاً إلى المسار الذي جرى اختياره في خطوة تحديد مجلد التنزيل ومن ثم عن طريق browse قم بتحديد المسار الكامل واختر الملف 'MATLAB2012b_std.dat' كما في الشكل



ستظهر اماك الواجهة التالية للدلالة على انتهاء التنزيل

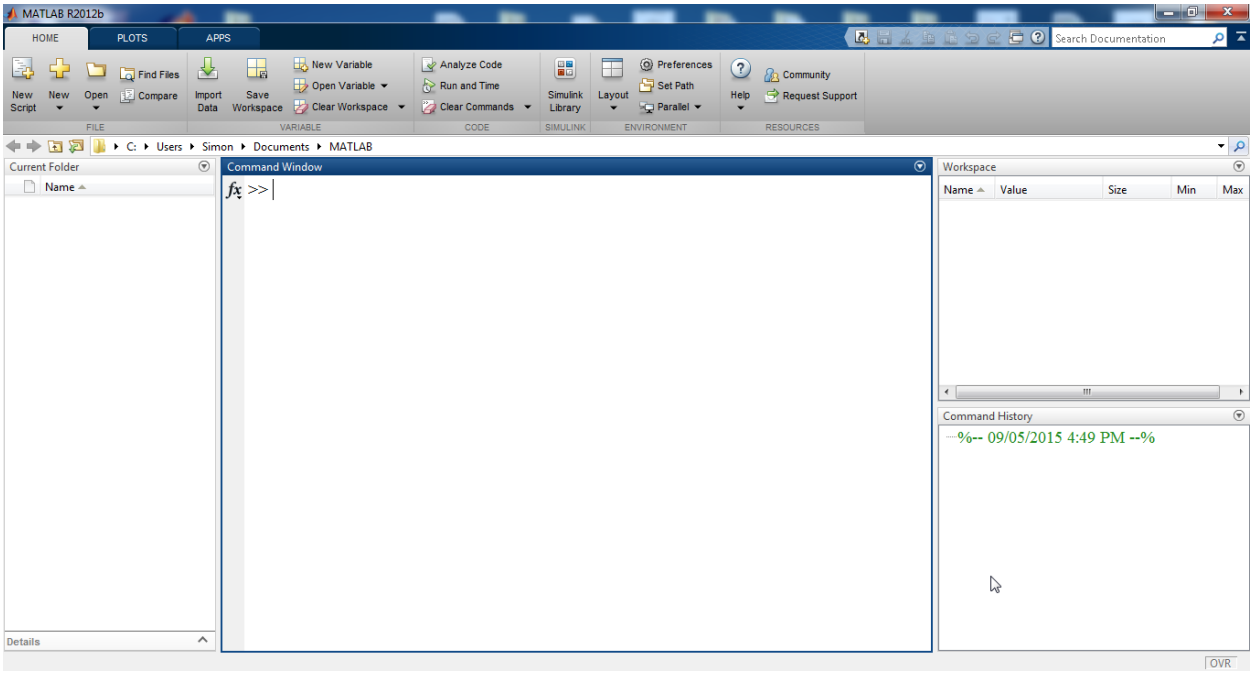


نتيجةً لوجود بعض الأخطاء في نسخة البرمجية يوجد ملف يجب إضافته موجود ضمن الملف الذي جرى فك ضغط أولاً وهو attachement اختر المجلد help سيفتح المجلد وضمنه مجلد اسمه includes سيفتح المجلد وضمنه مجلد اسمه product سيفتح المجلد وضمنه مجلد اسمه scripts ستجد ملف يدعى localnav قم بنسخ هذا الملف إلى مجلد التنزيل، إلى المسار السابق نفسه help\includes\product\scripts بهذا يكون قد اكتمل تنصيب البرمجية.

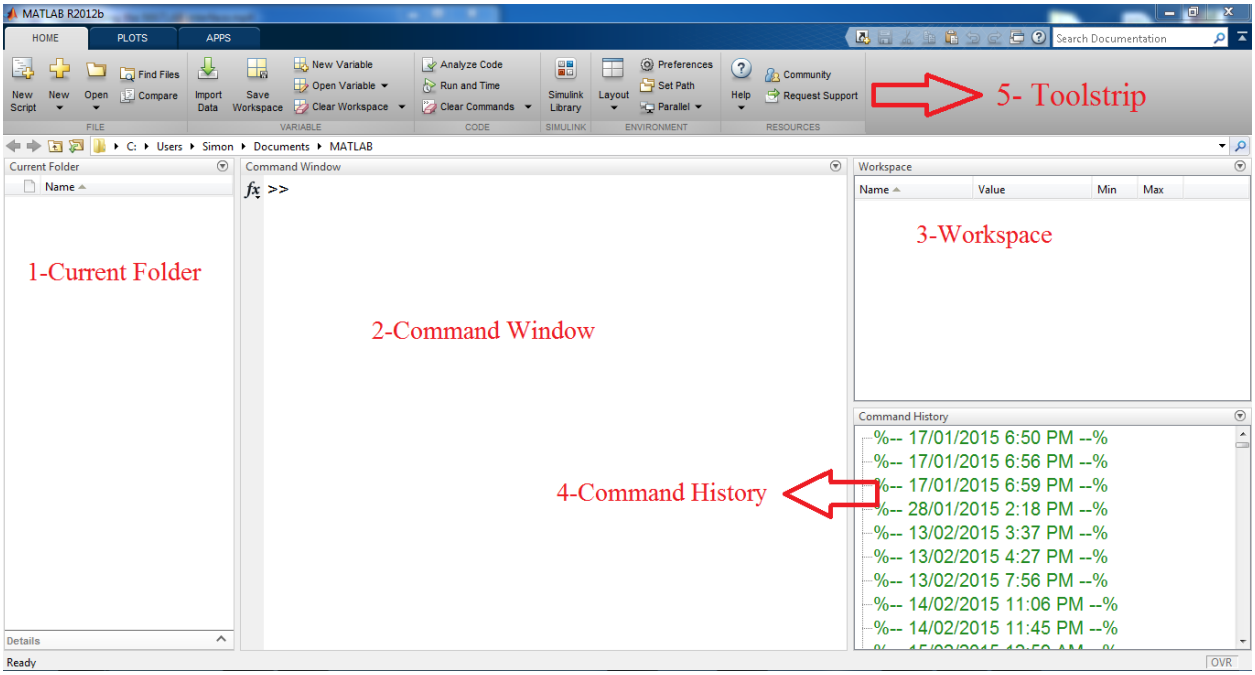
فهم الواجهة التخاطبية لبرمجية MATLAB

سنقوم الآن بالتعرف على الواجهة التخاطبية لـMATLAB، عند تشغيل البرمجية وذلك بعد النقر مرتين على الأيقونة التالية الموجودة على سطح المكتب بزر الفأرة mouse اليساري، أو عن طريق تحديد الأيقونة التالية ثم ضغط الزر اليميني واختيار "فتح أو Open"

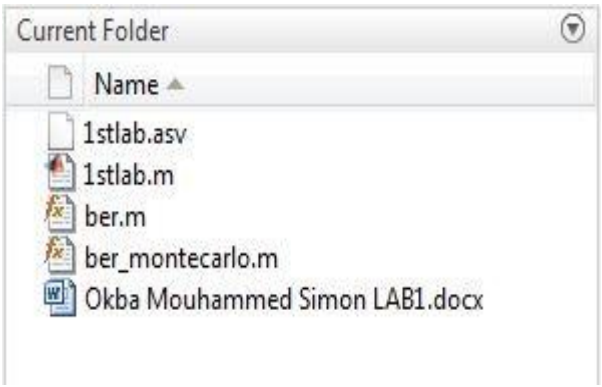
ستظهر أمامك الواجهة التالية



ملاحظة في حال استخدام نسخة أحدث من النسخة الحالية، وهي R2012b، أو أقدم ليست مشكلة كبيرة فالاختلاف يكون بالإظهار لكن الأجزاء الرئيسية تبقى ذاتها في كافة الإصدارات. الواجهة السابقة مقسمة إلى خمسة أجزاء رئيسية وهي موضحة بالصورة التالية



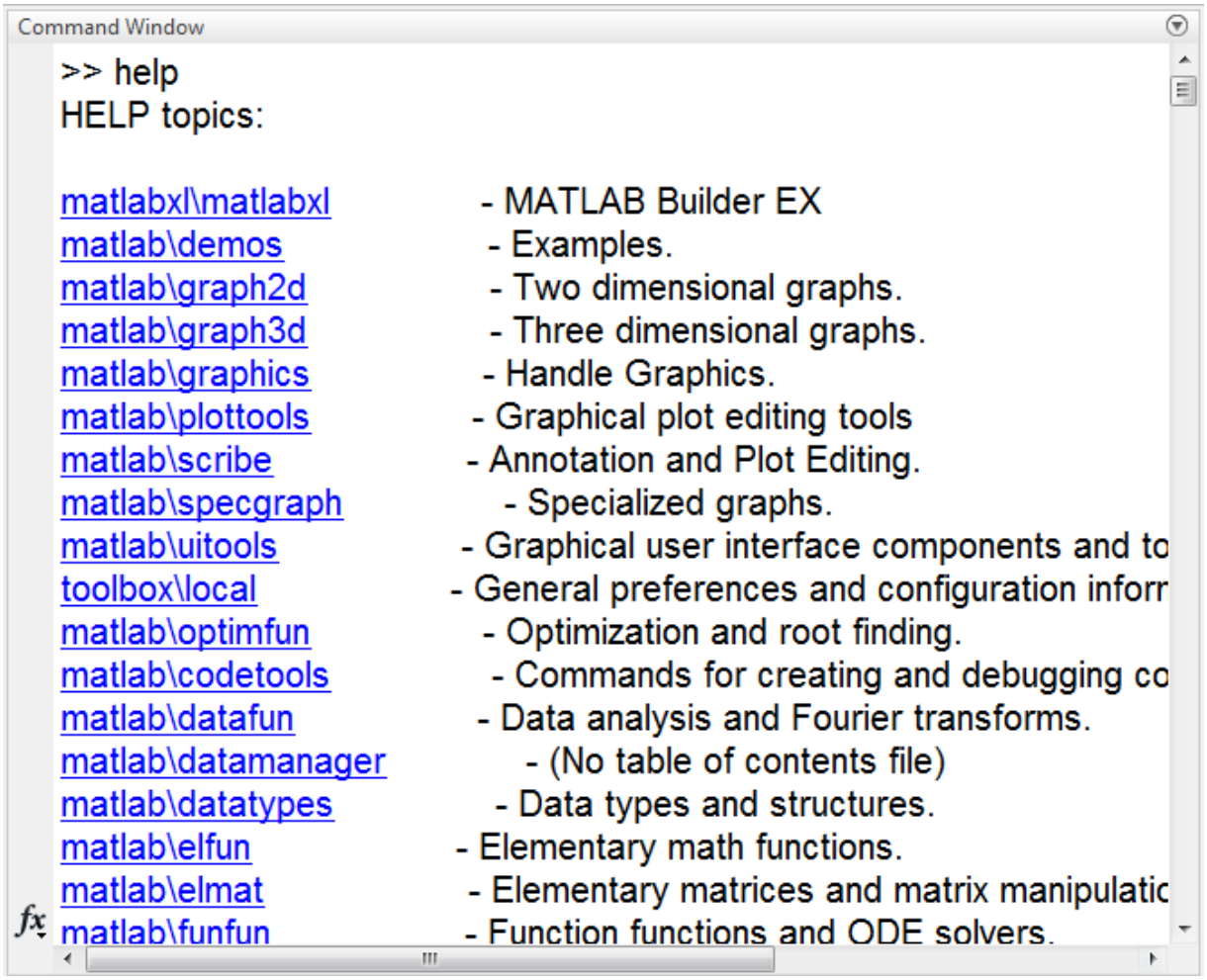
1. Current Folder وهو يعرض جميع الملفات المتعلقة بـMATLAB حيث تم تخزين جميع التوابع، scripts، والملفات الأخرى بلواحق مثل asv. و m. المتعلقة بالبرمجية وذلك وفقاً لمسار العمل الحالي



2. Command Window تعمل هذه النافذة على أنها نافذة لعرض الدخل input والخرج output عند تنفيذ التعليمات Commands أو التوابع Functions المختلفة.

مثال:

اكتب help ضمن هذه النافذة ثم اضغط enter سيتم استعراض، كما في الشكل، جميع المواضيع الممكن إيجاد مساعدة حولها



- عند تنفيذ run أي تعليمة Command أو مجموعة تعليمات موجودة ضمن script (في فصل لاحق سيتم شرح script) يتم إظهار الخرج ضمن هذه النافذة.
(فكرة مهمة جداً جداً)
- يمكن منع إظهار الخرج ضمن Command window عن طريق إضافة " ; " بعد التعليمة أو المتحول، تفيد هذه التعليمة كثيراً في حالة الرمازات البرمجية الكبيرة وتنجيز الخوارزميات حيث الاهتمام فقط بالخرج النهائي وليس بخرج كل سطر

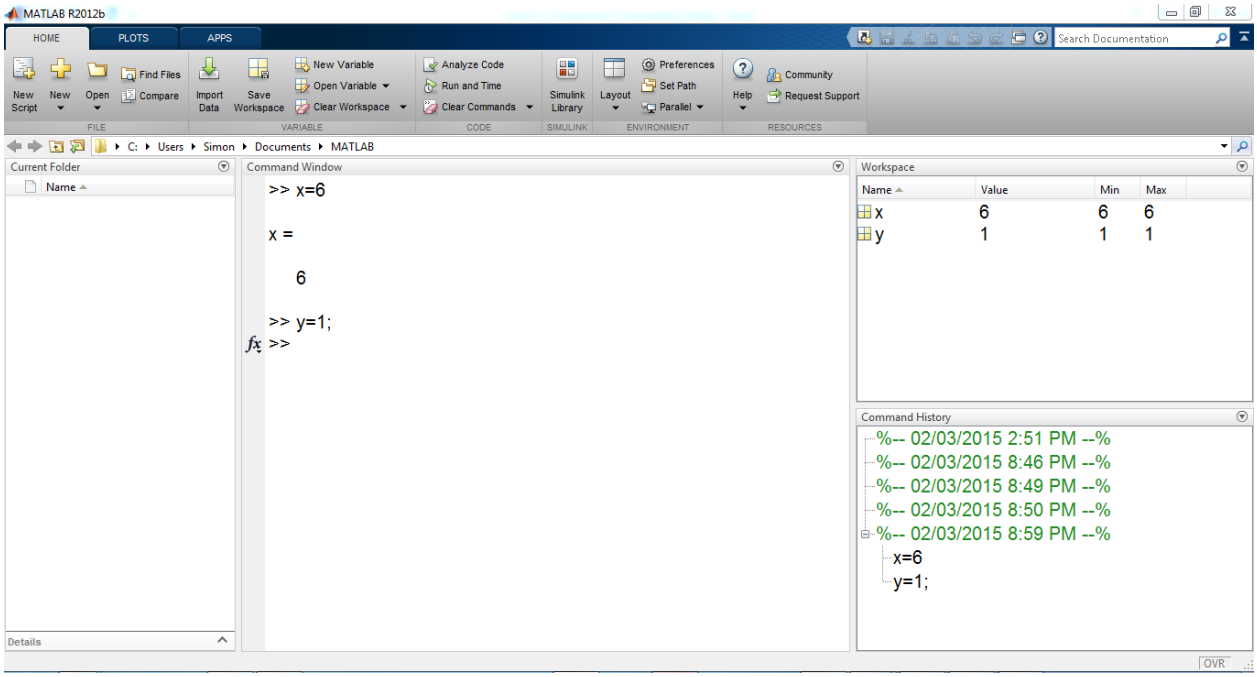
مثال على الفكرة السابقة

قم بتعريف المتحولx على الشكل x=6 ثم اضغط enter

ستلاحظ إظهار قيمة المتحول

ثم قم بتعريف المتحول yعلى الشكل y=1; ثم اضغط enter

ستلاحظ أن المتحول ظهر ضمن workspace دون ان يظهر ضمن command window

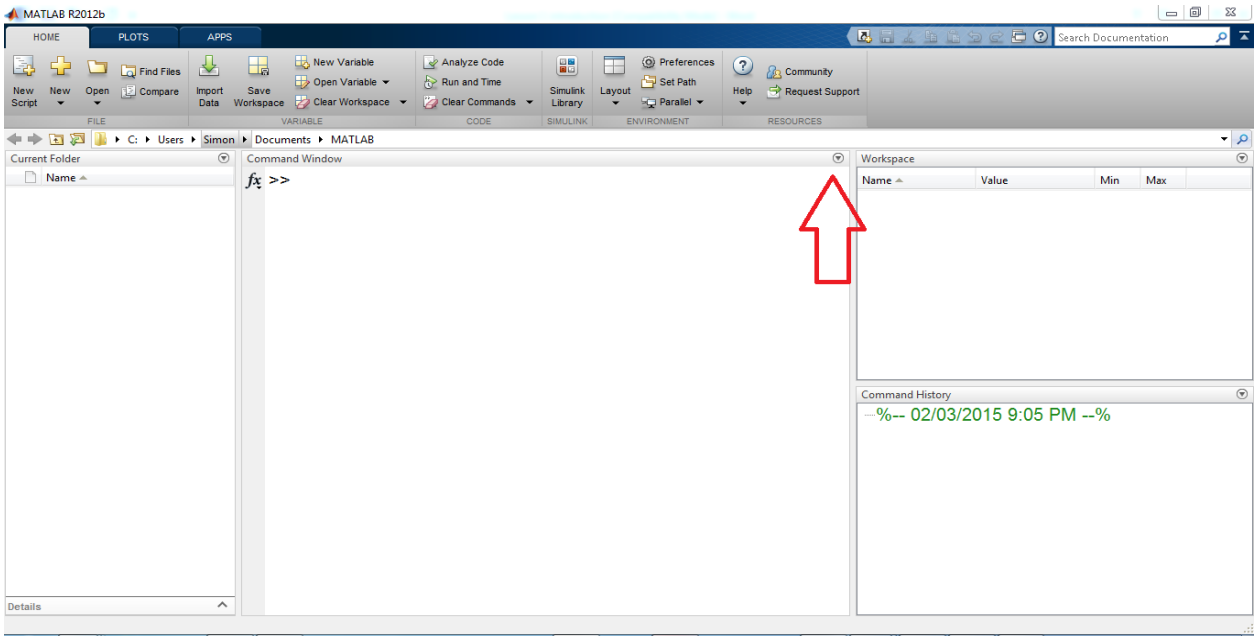


- يتم مسح جميع البيانات والتعليمات الموجودة ضمن Command Window فقط عن طريق تعليمة clc أما المتحولات لا يتم مسحها من فضاء العمل Workspace.

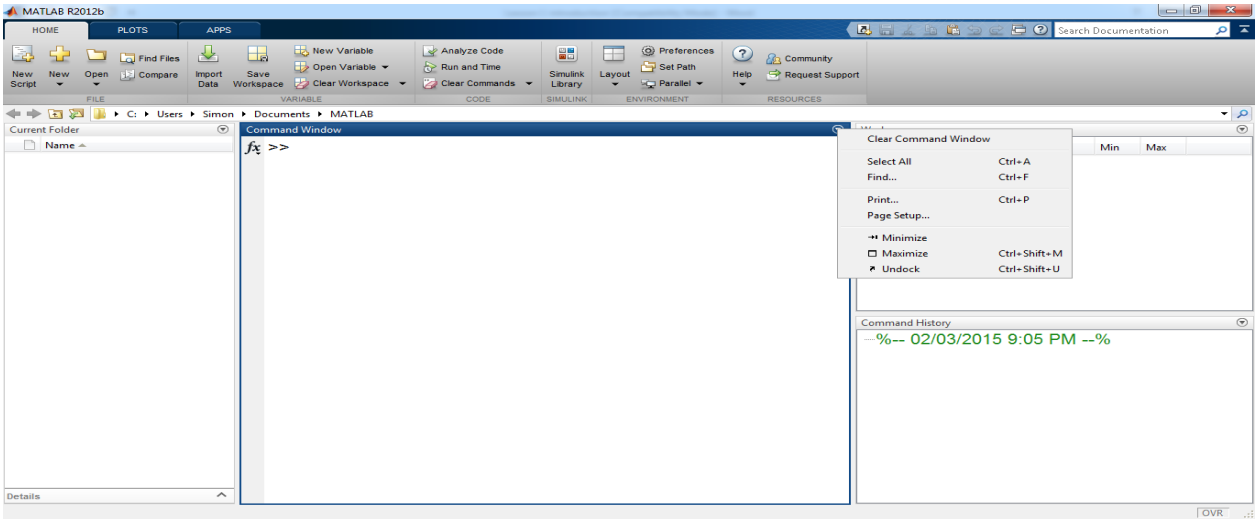
مثال

اضغط على help clc لمعرفة المزيد عن هذه التعليمة ثم enter

أو كما في الشكل التالي اضغط على الزر التالي



فيظهر الخيارات التالية

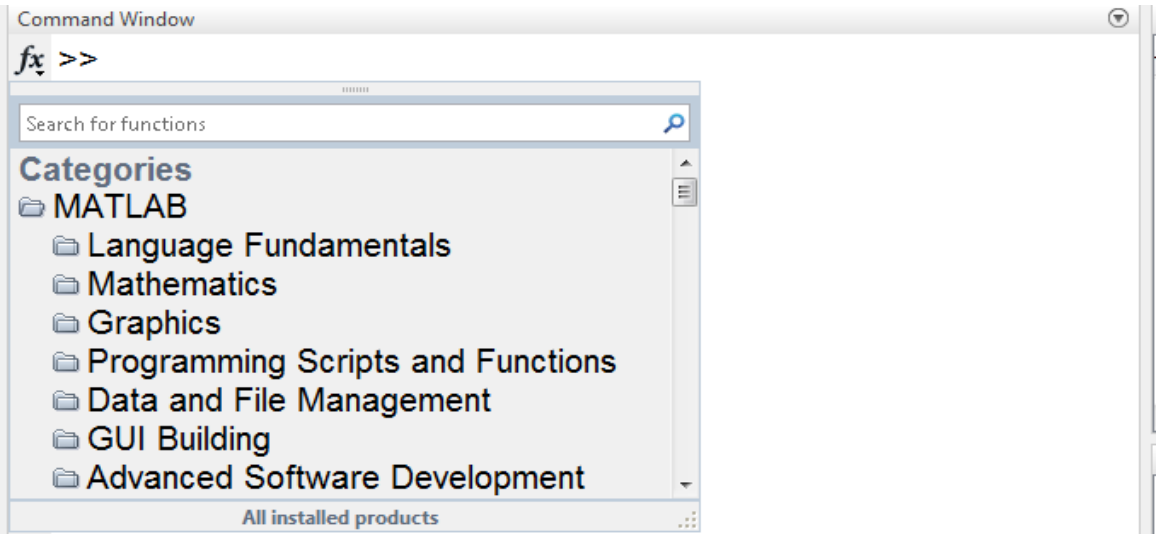


اختر clear command window.

- عند الكتابة ضمن command window يمكن استخدام زر Esc و استخدام Ctrl +C من أجل
  - Esc مسح التعليمة ضمن السطر الحالي (used for clear command line)
  - Ctrl+C الخروج من العملية الحالية والانتقال إلى سطر جديد من ال command line.

(used for Quit current operation and return control to the command line)

- يمكن أيضاً الضغط على الرمز $f(x)$ الموجود ضمن command window من أجل الحصول على مساعدة help حول التوابع الموجودة ضمن البرمجية



- يمكن الاستفادة من command window أيضاً للحصول على Help و Documentation حيث ان جميع توابع MATLAB تدعم التوثيق Documentation

تتضمن الملفات الموثقة على امثلة لاستخدام التوابع إضافةً لتوصيف التابع بشكل مفصل من حيث الدخل والخرج و calling syntax اي القواعد (النحو) للاستدعاء.

من اجل فتح نافذة منفصلة تحتوي على function documentation يمكن استخدام التعليمة doc ضمن الcommand window على الشكل

```
doc mean
```

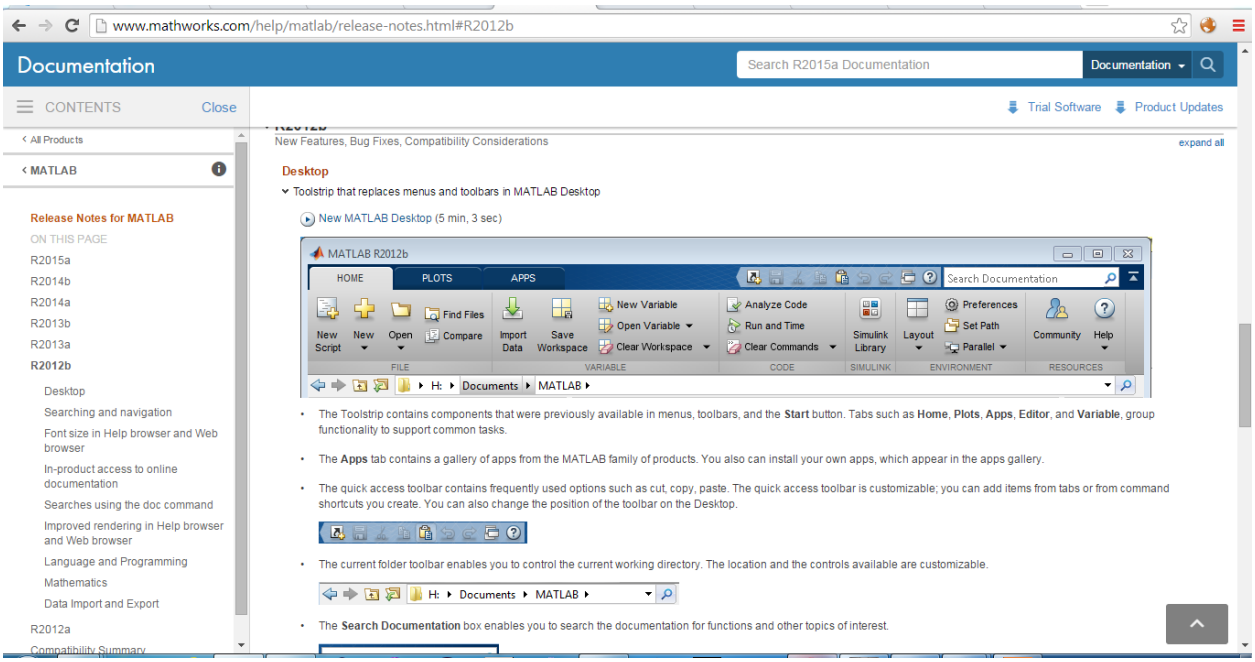
من أجل عرض مساعدة وتنويه hint حول التابع، يمكن كتابة اسم التابع في command window ثم فتح قوس من أجل بارمترات دخل التابع والانتظار على الشكل التالي

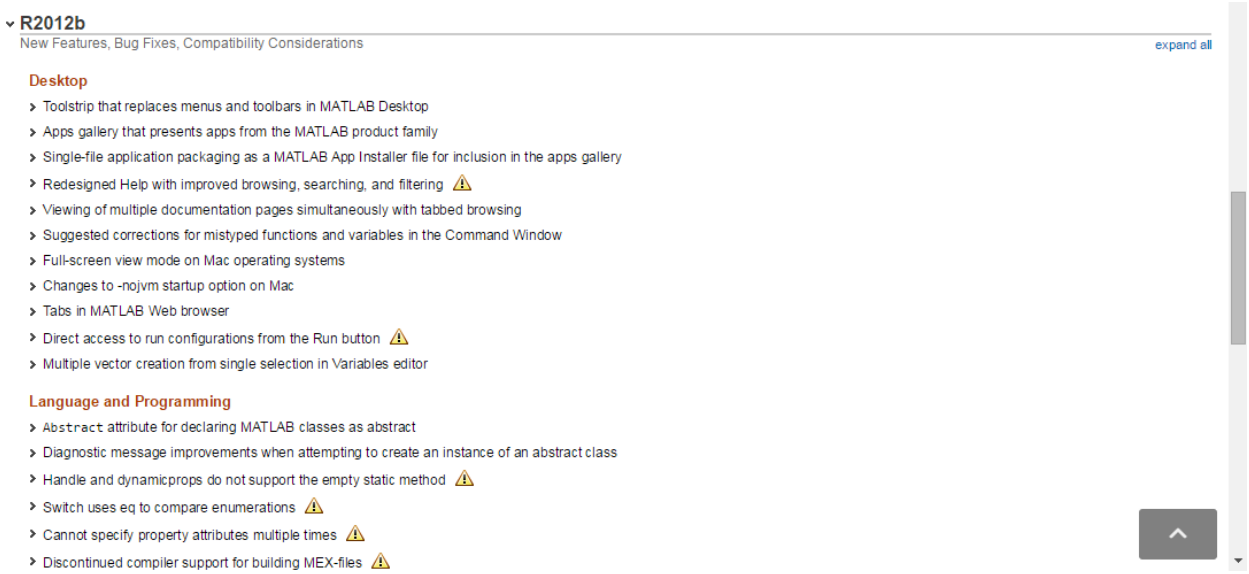
```
mean(
```

سيتم تقديم مساعدة مفيدة بعد العملية السابقة

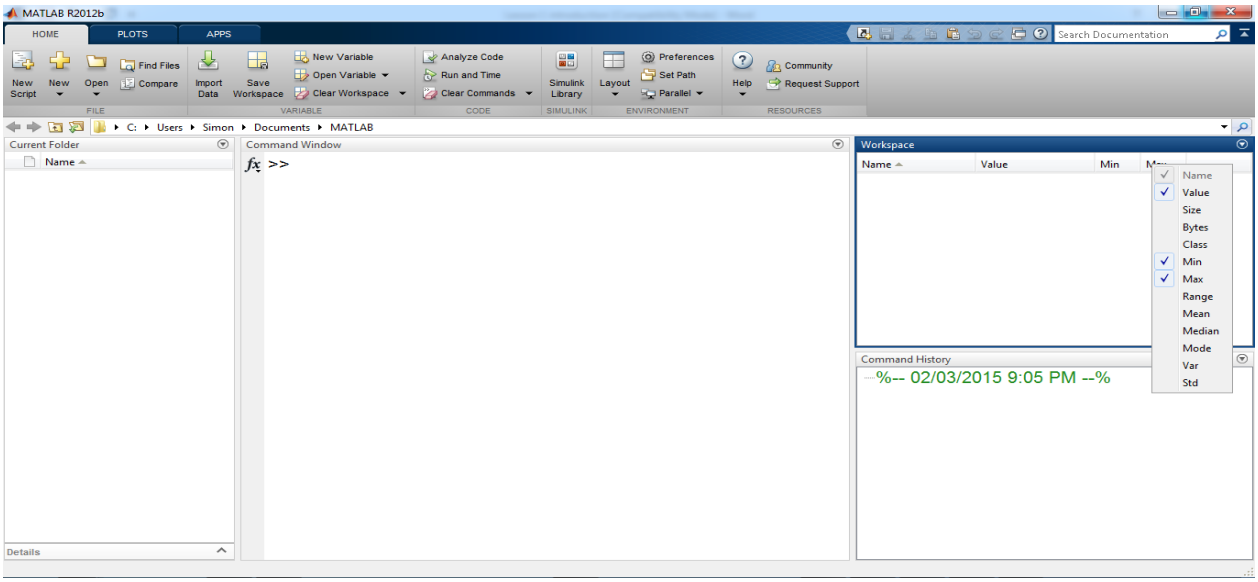
كما نذكر بإمكانية استخدام تعليمة help ضمن command window على الشكل التالي

```
help mean
```





3. Workspace يتم في هذه النافذة عرض أسماء المتحولات التي جرى تعريفها، قيمتها، القيمة الصغرى Min، القيمة الكبرى Max، إضافةً لخيارات إضافية يمكن استعراضها كما في الشكل التالي، عن طريق الضغط بالزر اليميني على الشريط، فتظهر الخيارات وهي (Name, Value, …, Max)



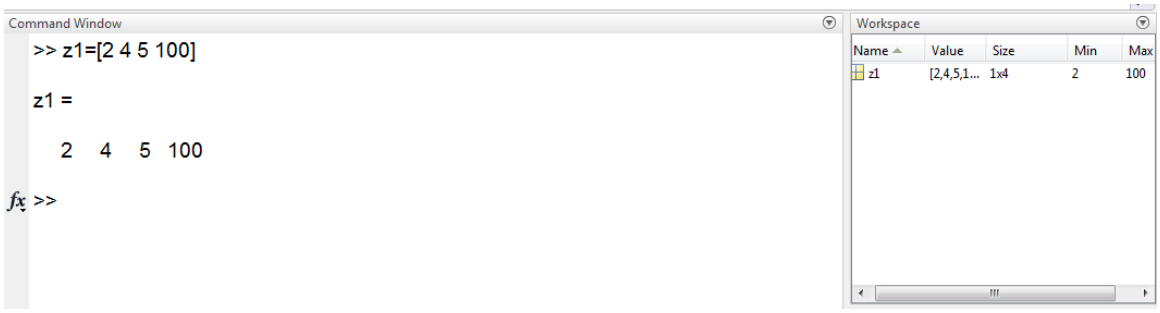
- لننفذ المثال التالي

ضمن command window اكتب z=2 ثم enter

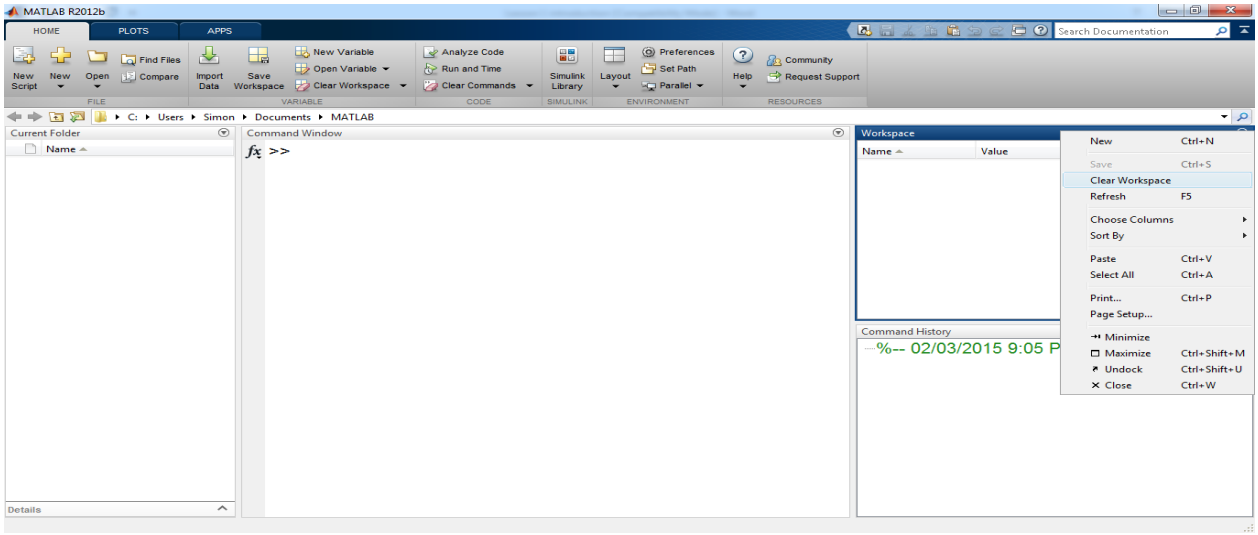
يتم تعريف متحول هو عبارة عن مصفوفة أبعادها 1 × 1 أي عبارة عن قيمة سلميّة scalar

ضمن command window اكتب z1=[2 4 5 100] ثم enter

يظهر لك التالي حيث أن المتحول هو عبارة عن شعاع بعده 4 × 1 نلاحظ كيف تم إيجاد القيمة الصغرى والكبرى تلقائياً



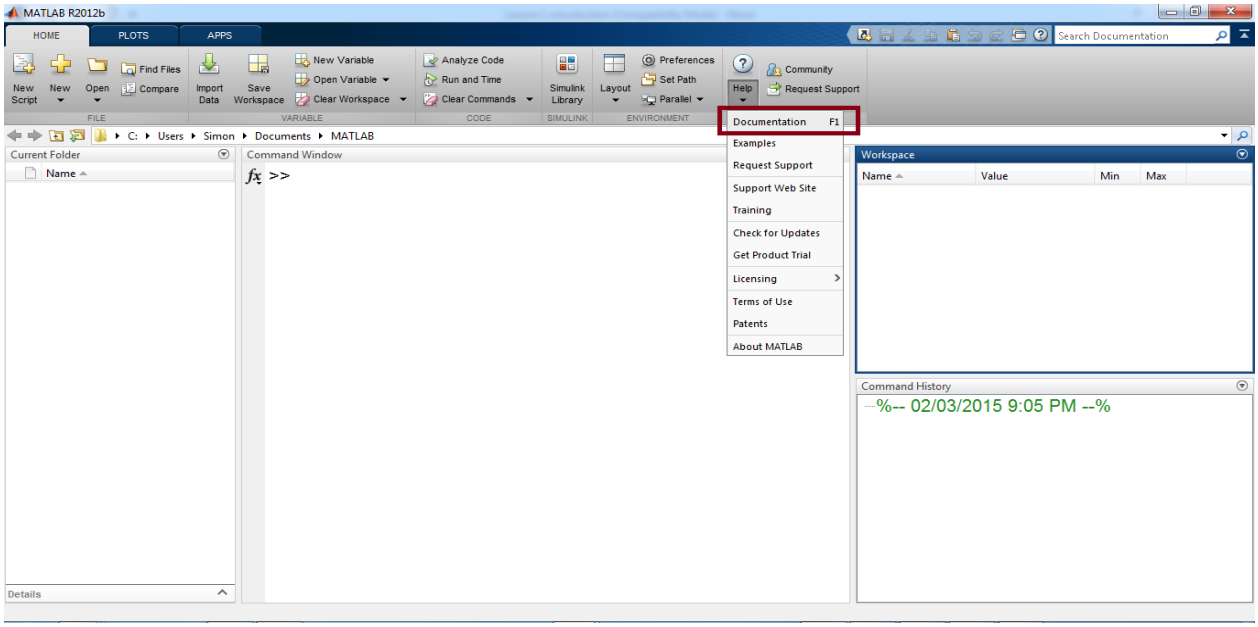
- كما يمكن مسح المتحولات المعرفة ضمن workspace عن طريق خيار clear workspaceكما في الشكل التالي



أو عن طريق تعليمة clear

اضغط على help clear لمعرفة المزيد عن هذه التعليمة وإضافةً للفرق مع clear all

أو يمكن فتح help كما في الصورة

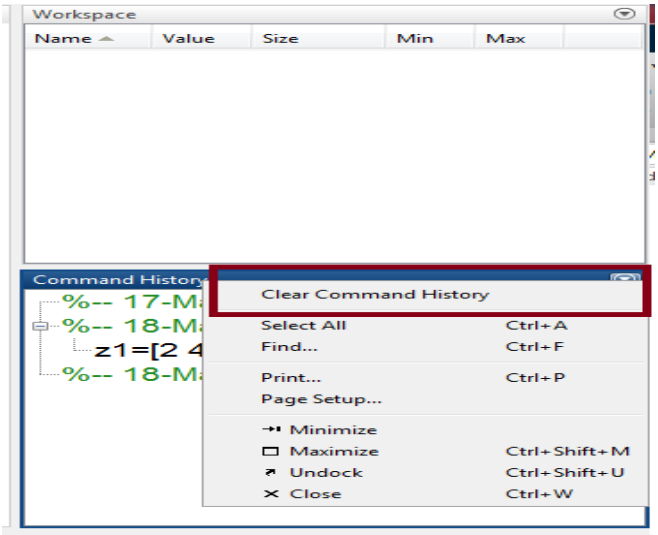


وكتابة clear للبحث عن التعليمة وفهم عملها مع أمثلة عنها وكذلك الحال من أجل أي تعليمة أو تابع.

4. Command History تحتفظ بنسخة من جميع التعليمات التي تم تنفيذها إضافةً إلى تاريخ التنفيذ مما يتيح لنا العودة لمعرفة المتحولات والتوابع التي جرى تعريفها مسبقاً،

يمكن أيضاً إعادة تنفيذ التعليمات أو إعادة تعريف المتحولات التي جرى استخدامها سابقاً عن طريق الضغط عليها فقط.

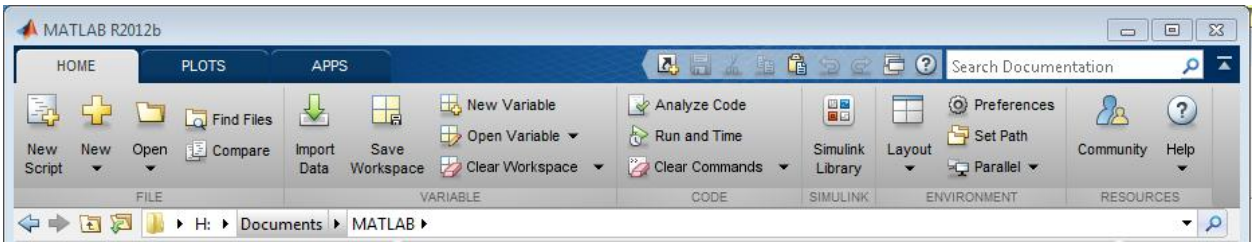
يمكن مسح command history عن طريق الضغط ضمن النافذة بالزر اليميني فيظهر عدد من الخيارات ضمن هذه الخيارات Clear Command History كما في الشكل



5. Toolstrip (الجزء الخامس هذا toolstrip هو جزء جديد خاص بالنسخ vesion 8 والنسخ اللاحقة)

بعد التعرف على الواجهة التخاطبية، سننتقل للتعرف على محتويات الشريط العلوي أو ما يسمى toolstrip سنهتم حالياً بالجزء الاول وهو Home أما الجزء الثاني فهو Plots سيتم التعرف عليه لاحقاً وهو يحتوي

على توابع لإظهار منحنيات وأشكال ثنائية وثلاثية الأبعاد والجزء الثالث apps سيتم شرحه لاحقاً. يبين الشكل التالي صورة ل toolstrip.



تعتبر Toolstrip إحدى أهم التعديلات التي طرأت على MATLAB في النسخة الثامنة Version 8 من حيث شكل البرمجية فهي تعتبر الطريقة الجديدة في MATLAB للحصول على الخصائص الوظيفية ضمن البرمجية ، حيث قامت Toolstrip بتجميع كافة الخاصائص الموجودة ضمن القائمة و شريط الأدوات Menus & Toolbars الموجودين في النسخ السابقة.

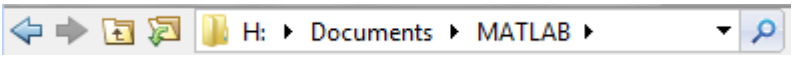
تحتوي Toolstrip على عناصر كانت موجودة مسبقاً في menus, toolbars و زر start وهية تحتوي على عدة tabs وهي Home, Apps, Editor and Variable إضافةً إلى مجموعات وظيفية تدعم خدمات شائعة الاستخدام.

يحتوي toolstrip على

a. شريط quick access

الذي يحتوي على عدة خيارات تستخدم بشكل متكرر مثل cut copy paste كما يمكن إضافة خصائص أخرى لهذا الشريط من الخيارات المتاحة في toolstrip ويمكن أيضاً تغيير موضعه.

b. شريط current folder الذي يتيح لك التحكم بمسار العمل الحالي.



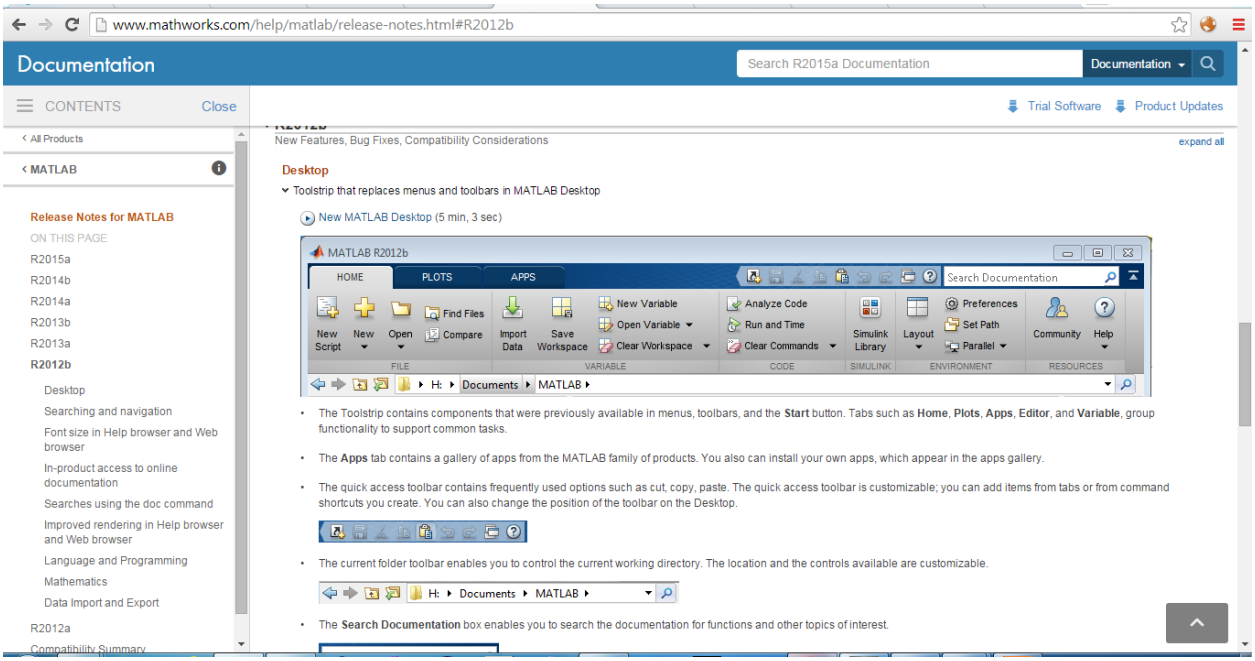
c. Search Documentation box يتيح للمستخدم البحث عن الملفات التي جرى توثيقها عن التوابع والتعليمات.

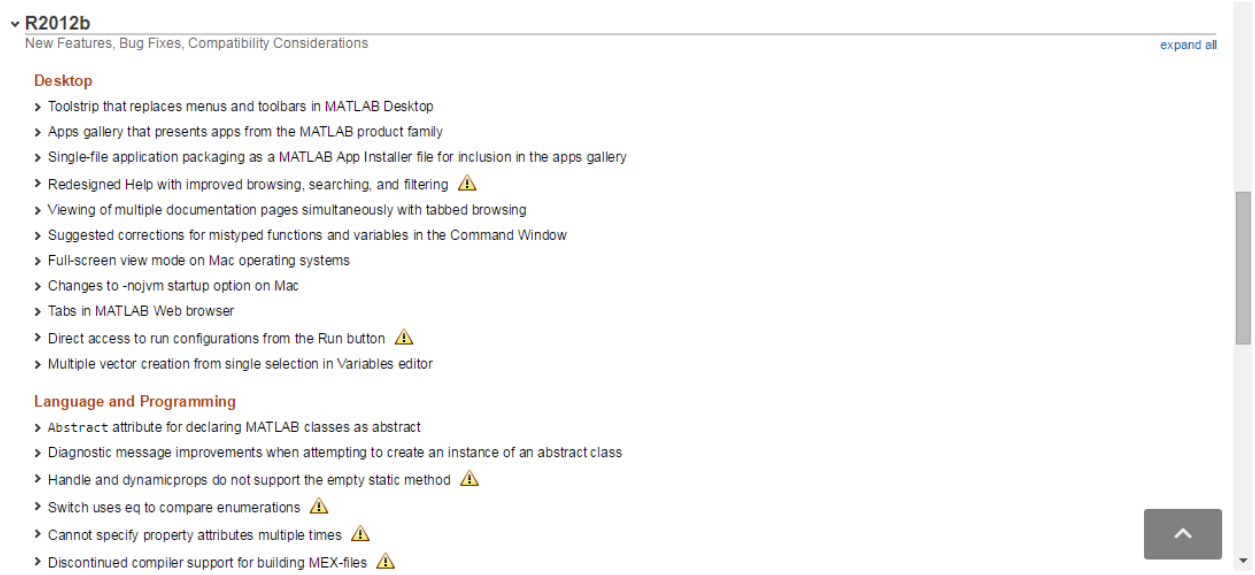
ننتقل الآن إلى تقديم شرح عن قسم Home

يتألف من عدة أجزاء تتعلق بوظائف مختلفة، وهي

- File من أجل إنشاء scripts, function, Simulink models,....
- Variable للتحكم بالمتحولات من حيث إنشاء متحول جديد، مسح بعض المتحولات إضافةً لإمكانية استيراد المعطيات import data.

- Code من اجل تنفيذ التعليمات والرمازات البرمجية
- Simulink من اجل الوصول إلى مكاتب Simulink
- Environment تحتوي على Layout و preferences من اجل تغيير شكل البرمجية من ناحية ما يتم عرضه ضمن المواجهة البيانية وضبط الاعدادات الخاصة بالعرض مثل نوع الخط وحجمه، إضافة الوان للمتحولات و التعليمات و .....
- Resources تحتوي على help وتتيح لك الاتصال مع موقع الشركة

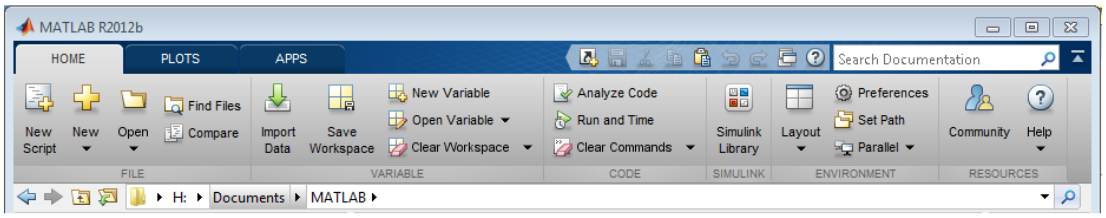




Desktop

Toolstrip that replaces menus and toolbars in MATLAB Desktop

New MATLAB Desktop (5 min, 3 sec)



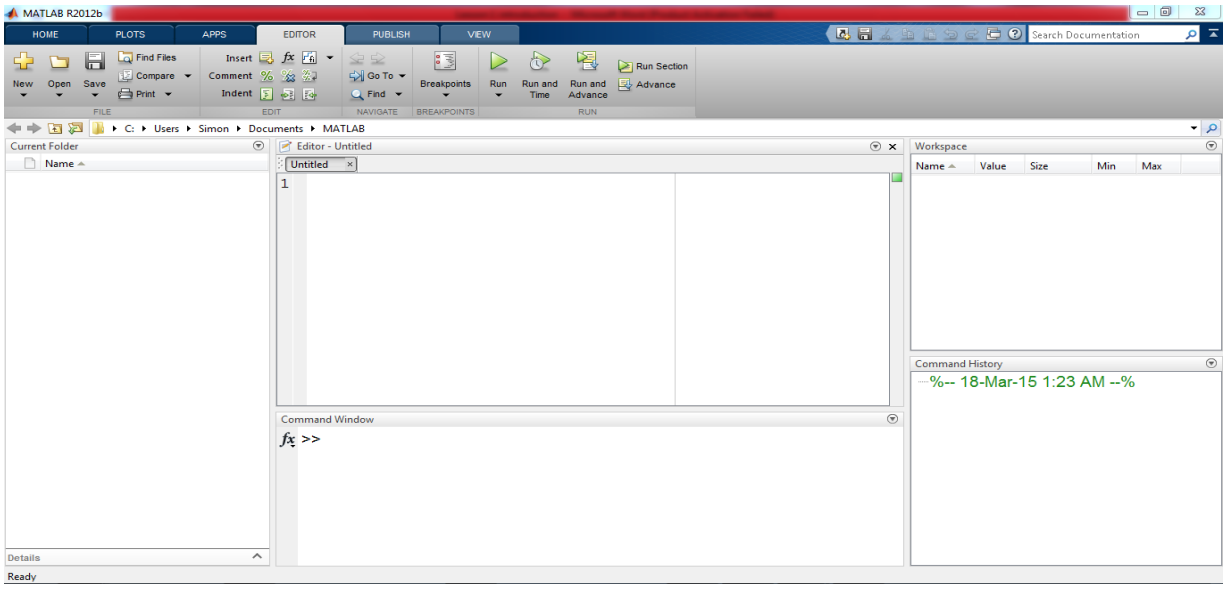
- The Toolstrip contains components that were previously available in menus, toolbars, and the **Start** button. Tabs such as **Home**, **Plots**, **Apps**, **Editor**, and **Variable**, group functionality to support common tasks.
- The **Apps** tab contains a gallery of apps from the MATLAB family of products. You also can install your own apps, which appear in the apps gallery.
- The quick access toolbar contains frequently used options such as cut, copy, paste. The quick access toolbar is customizable; you can add items from tabs or from command shortcuts you create. You can also change the position of the toolbar on the Desktop.

- The current folder toolbar enables you to control the current working directory. The location and the controls available are customizable.

- The **Search Documentation** box enables you to search the documentation for functions and other topics of interest.

a. File

- عند الضغط على New Script يتم إنشاء Scriptجديدة (سيتم شرح معنى script في الفصل القادم) كما في الشكل

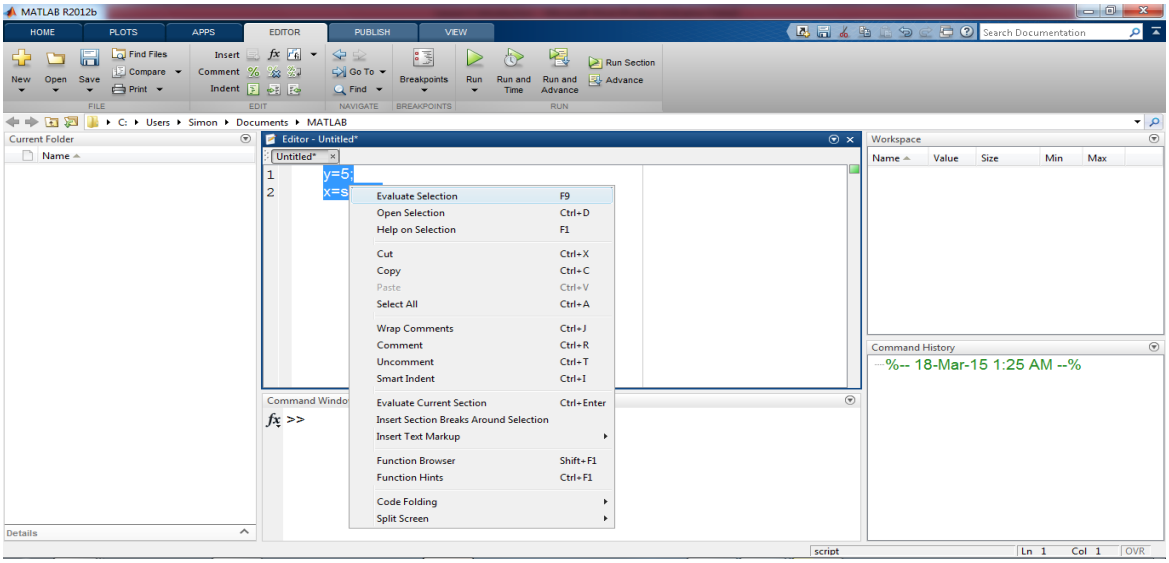


يمكن ضمن script تعريف متحولات

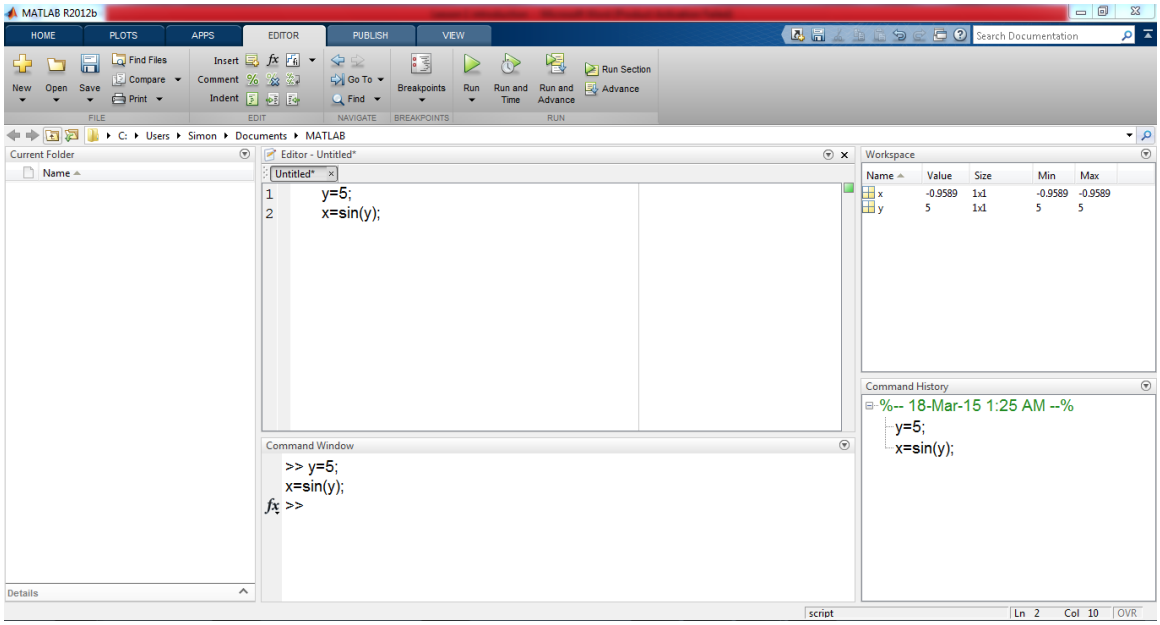
مثلاً

y=5; x=sin(y);

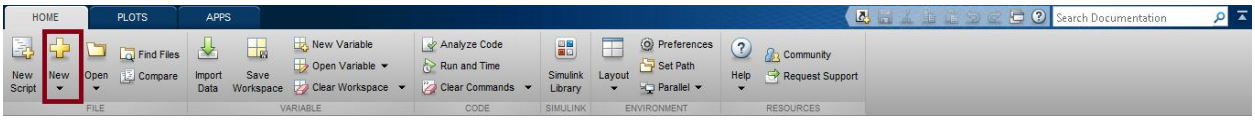
من أجل التنفيذ نضغط F9 أو نحدد كامل الرماز المكتوب ونضغط بالزر اليميني لنحصل على عدد من الخيارات نختار منها Evaluate Selection كما في الشكل



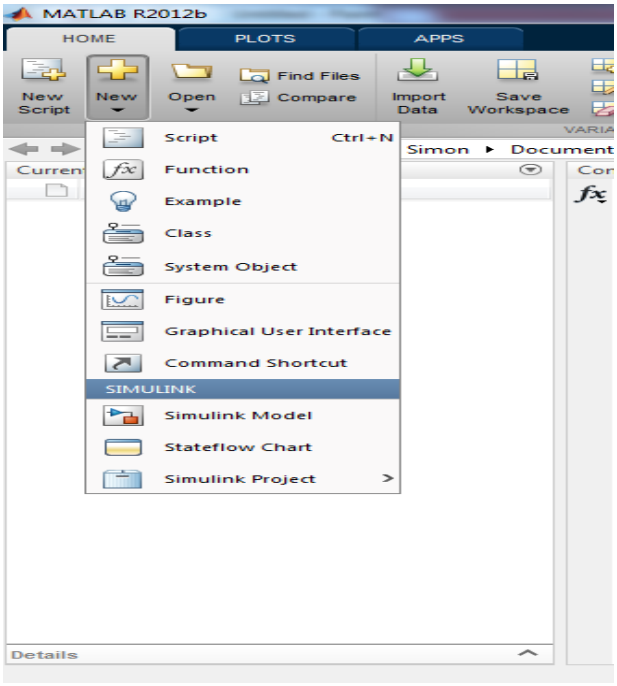
نقوم بعدها بالحصول على النتيجة ضمن workspace حيث جرى تخزين قيمة المتحولات



- من new نحصل على عدد كبير من الخيارات:

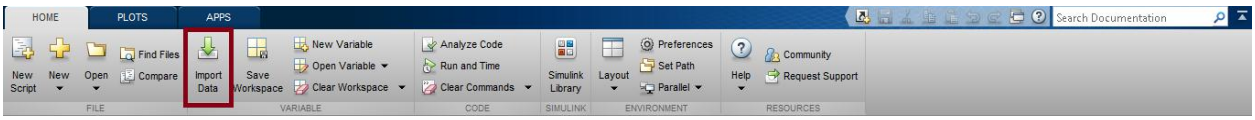


حيث يمكن تعريف script، تابع، class، شكل، واجهة بيانية تخاطبية GUI، ونموذج Simulink.

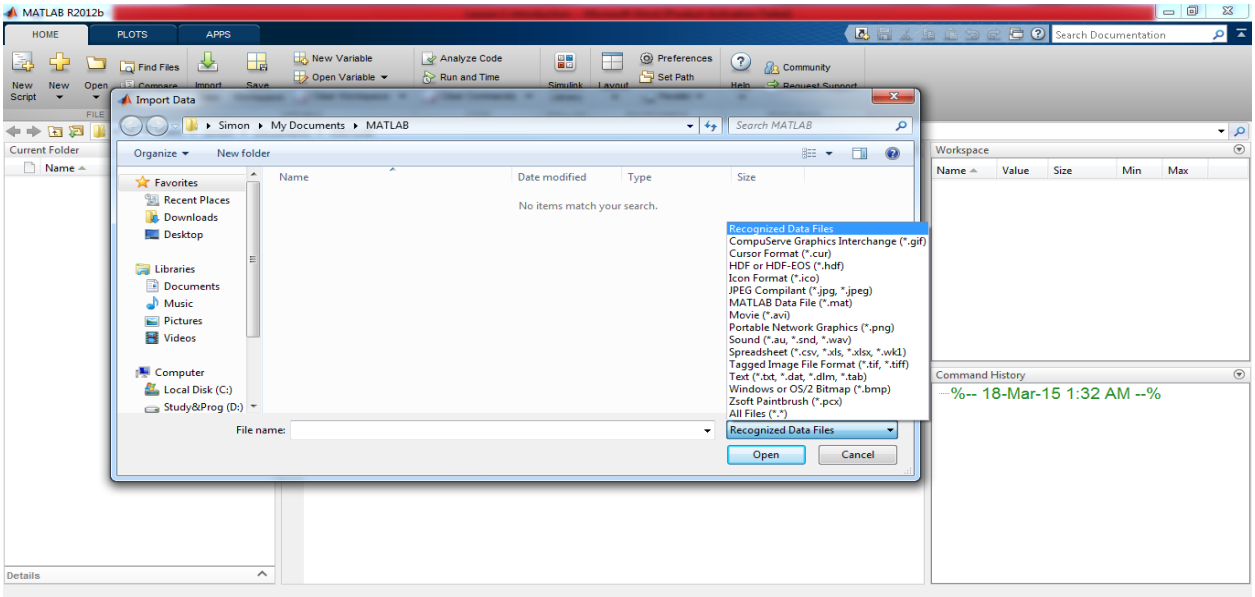


.b Variable

من الممكن الحصول على معطيات عن طريق Import Data



عند الضغط على الخيار import data ستفتح واجهة اخرى كما في الشكل

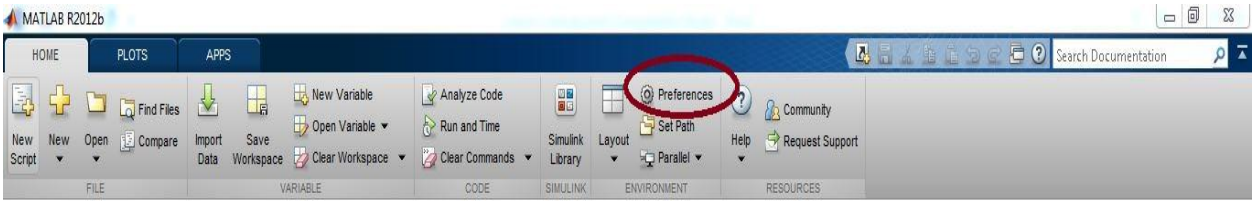


نلاحظ إمكانية استيراد المعطيات المخزنة بعدد صيغ سواء كانت صور، ملف نصي، فيديو أو ملف MATLAB.

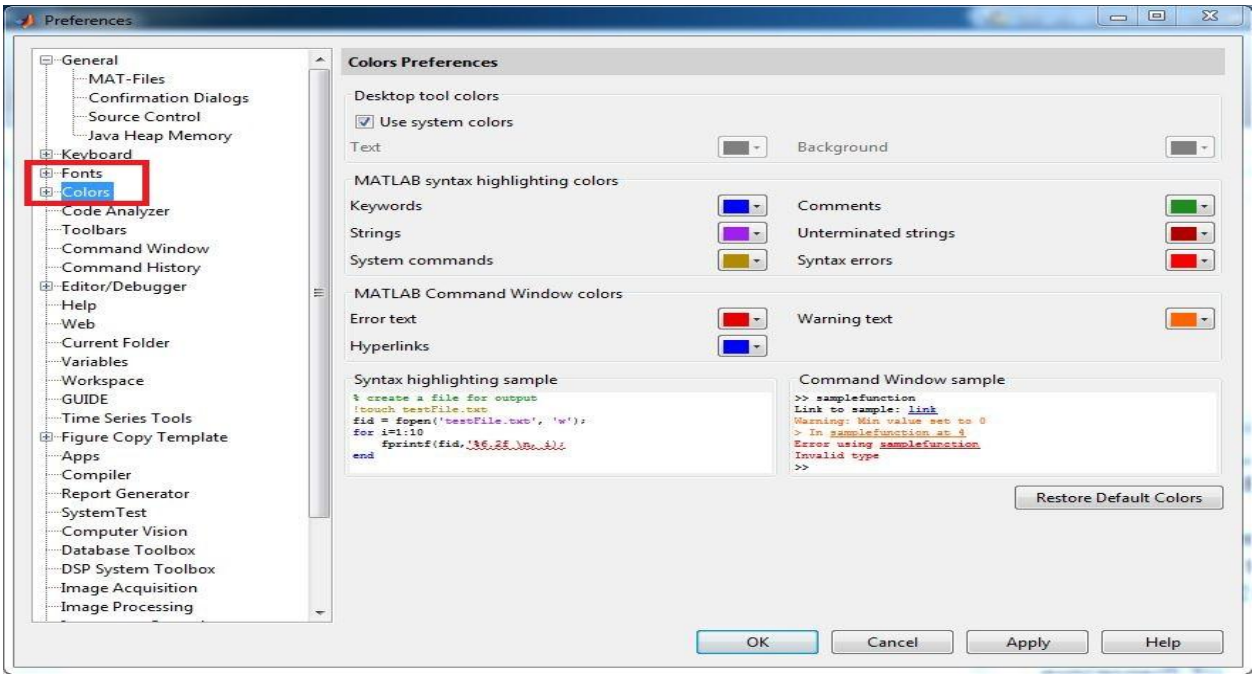
يمكن ايضاً إنشاء متحول جديد عن طريق NewVariable، مسح المتحولات التي جرى تعريفها سابقاً Clear أو نتائج الحسابات او فضاء العمل عن طريق Clear Workspac، ويمكن أيضاً معاينة قيم متحول تم تعريفه عن طريق Open Variable إضافةً لذلك يمكن حفظ workspace في حال الحاجة لاستخدامه لاحقاً عن طريق Save Workspace

c. Environment

من أجل ضبط الإعدادات والإظهار ضمن MATLAB ادخل إلى preferences



وقم بتغيير نوع الخط ضمن أي نافذة للبرمجية ولونه وحجمه مثلاً عن طريق fonts and colors

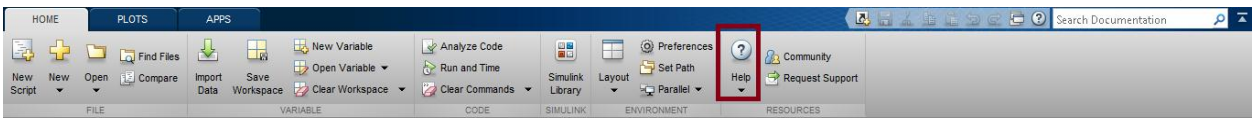


إضافةً لذلك تحتوي preferences على خواص عديدة ومتنوعة للتحم بعمل البرمجية وإظهاراتها

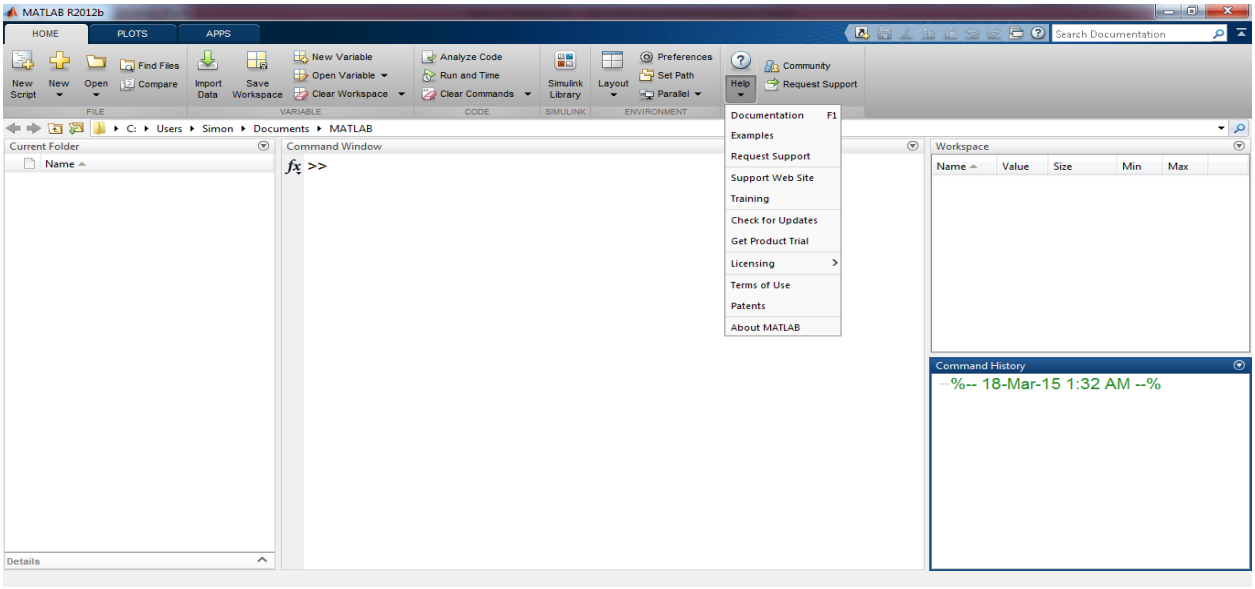
يمكن عن طريق Layout التحكم بالواجهات التي يتم عرضها حيث يمكن مثلاً إزالة Workspace او Current folder من الواجهة الرئيسية للبرمجية

d. Resources

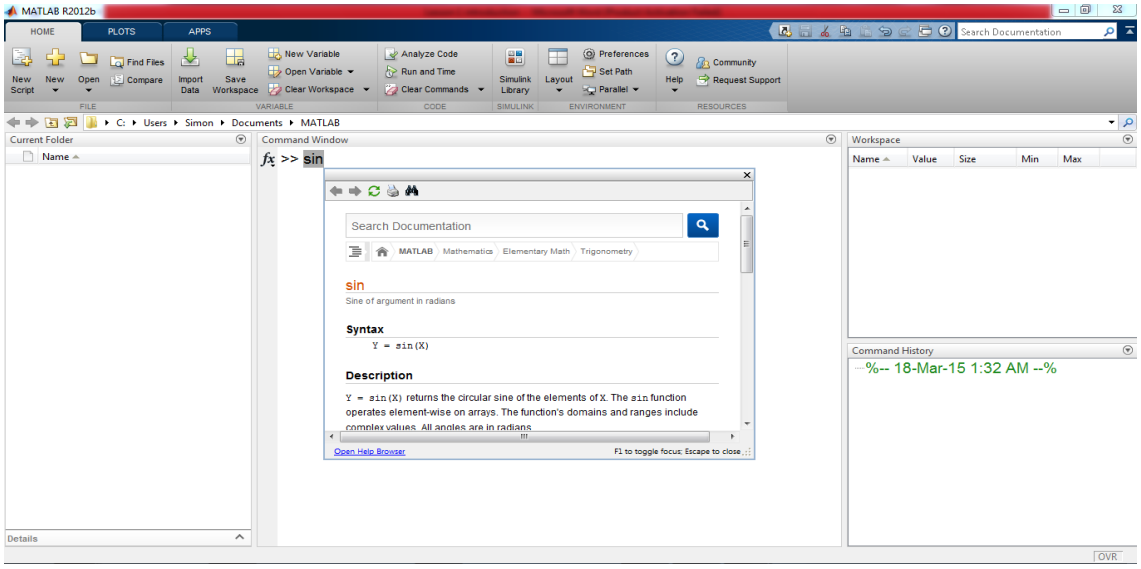
من أهم الأجزاء حيث يوجد help



ويمكن إيجاد المساعدة اللازمة عن الضغط عليها حيث يمكن فتح الملفات documentation أو الأمثلة إضافةً إلى العديد من الخيارات كما في الشكل



يمكن ايضاً الحصول على المساعدة help عن طريق كتابة التعلمية ثم الضفط على F1 كما في الشكل



ملاحظات هامة

1. يجب الانتباه إلى خطأ شائع كثيراً وذلك عند استعمال أوتنفيذ الرمازات البرمجية Codes التي جرى برمجتها وتخزينها مسبقاً،

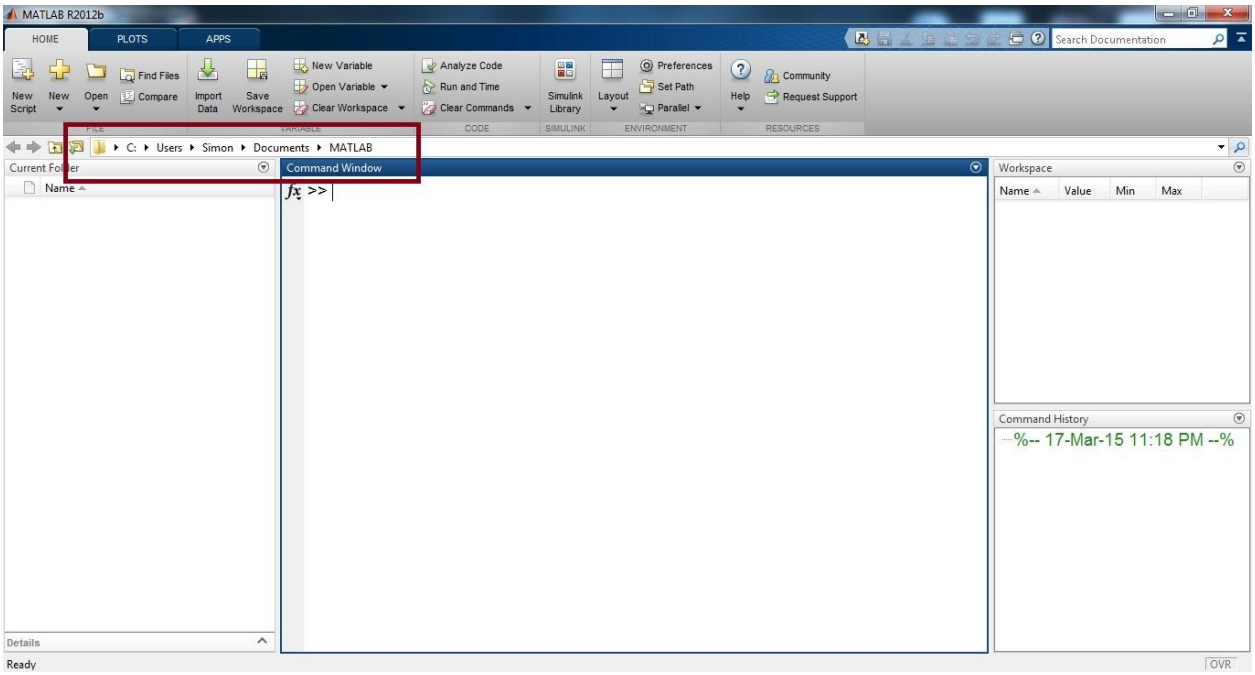
الخطأ هو: عند تنفيذ هذه التوابع يكون المسار الحالي Current path مختلف عن المسار الموجود ضمنه التابع.

الحل : يجب تغيير المسار الحالي إلى المسار الموجود ضمنه الملف الذي يجري تنفيذه وذلك باستخدام current folder toolbar

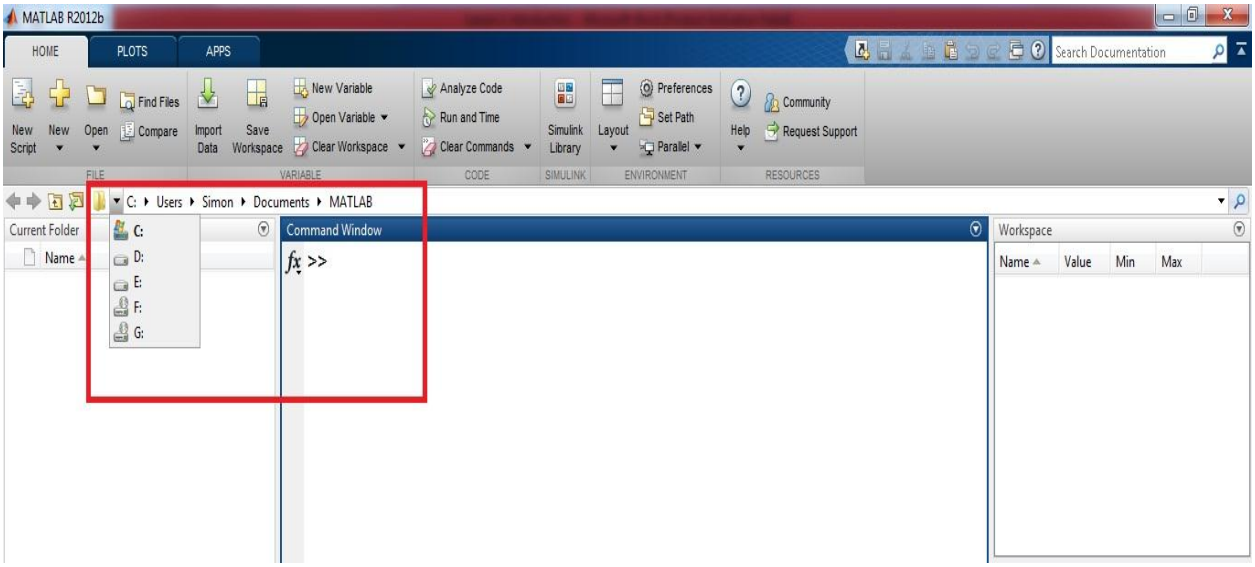
سنوضح كيفية تغيير فيما يلي:

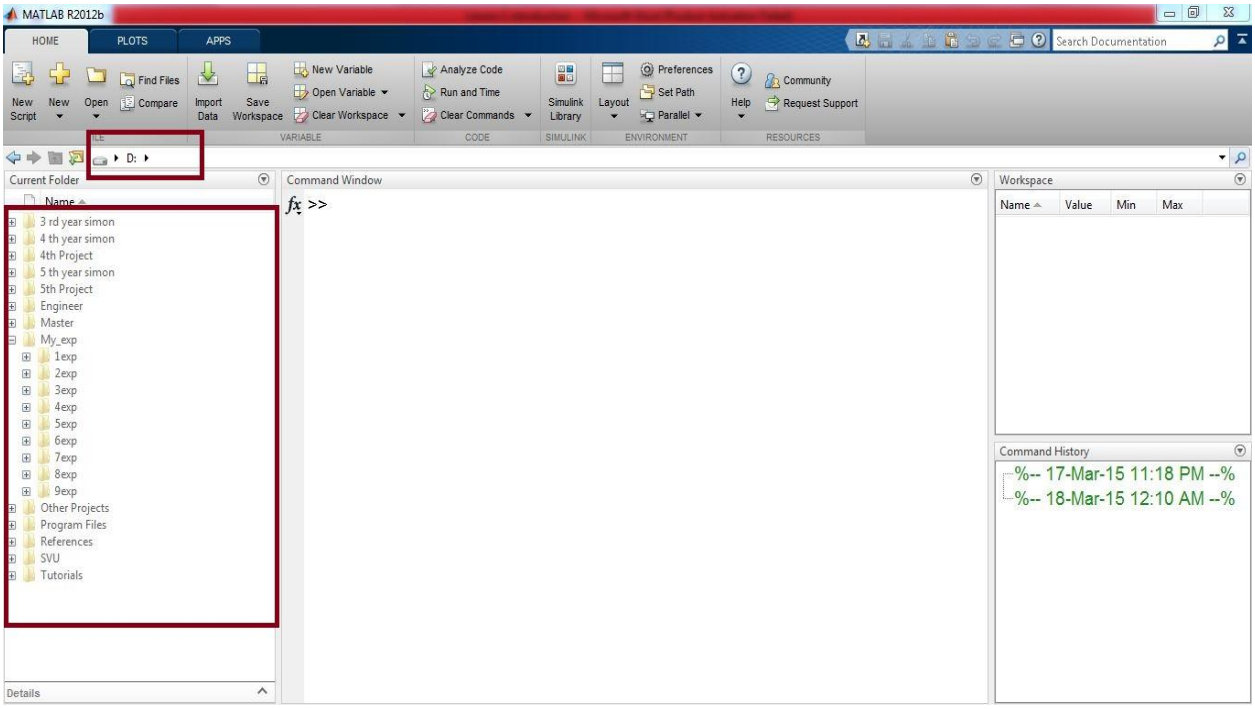
إن المسار الافتراضي default path لهذه البرمجية هو

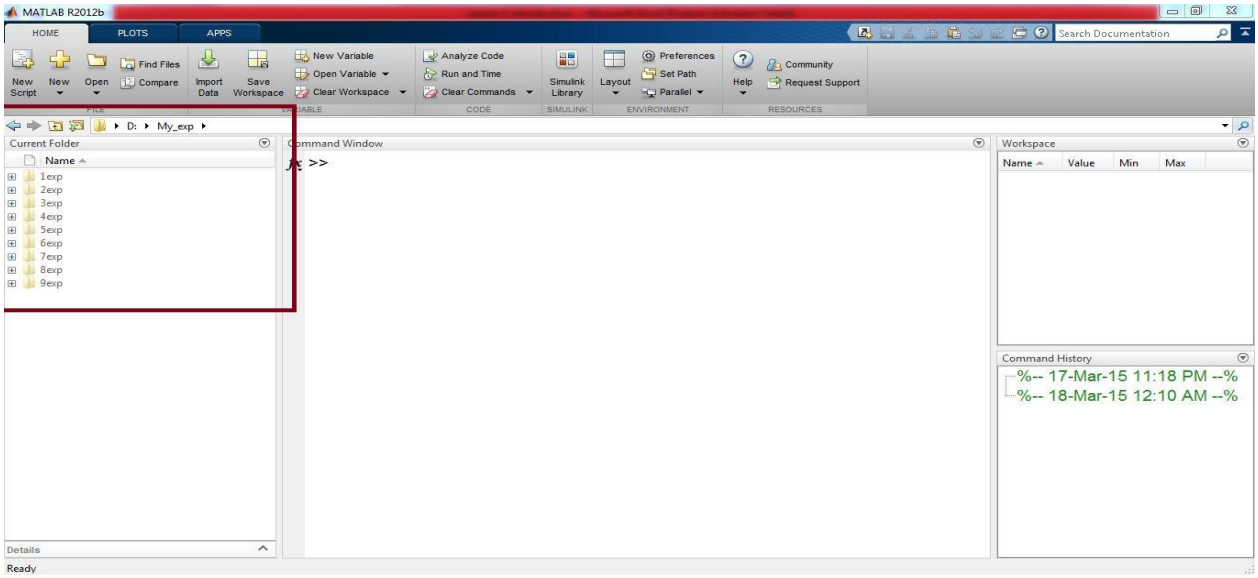
C: Users: yourComputerName: Document: MATLAB



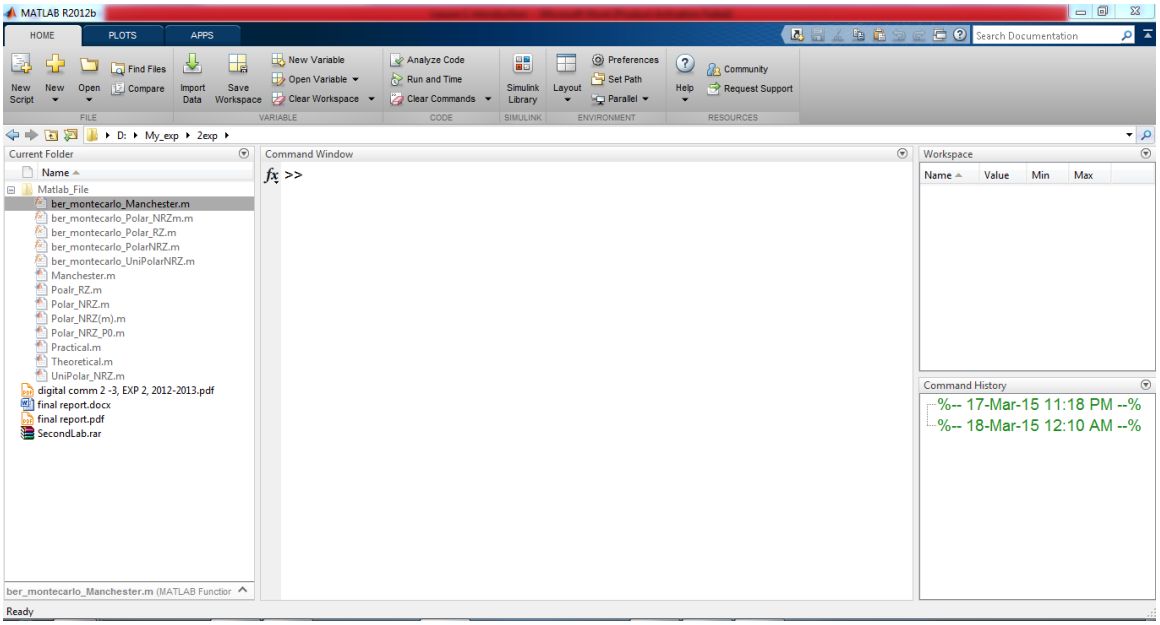
ويمكن تغييره بسهولة (نفترض أن الملفات المراد تنفيذها موجودة ضمن القرص D في مجلد اسمه My_exp) عن طريق الضغط على إشارة المجلد نجد أنه تم عرض الأقراص الموجودة ضمن الحاسب، نختار منها القرص المناسب ثم نختار الملف المناسب الذي يحتوي على البرامج المطلوب تنفيذها أو التي جرى تخزينها سابقاً


MATLAB R2012b
HOME
PLOTS
APPS
Search Documentation
New Script
New
Open
Find Files
Compare
Import Data
Save Workspace
New Variable
Open Variable
Clear Workspace
Analyze Code
Run and Time
Clear Commands
Simulink Library
Layout
Preferences
Set Path
Parallel
Help
Community
Request Support
FILE
VARIABLE
CODE
SIMULINK
ENVIRONMENT
RESOURCES
C: ▸ Users ▸ Simon ▸ Documents ▸ MATLAB
Current Folder
Name
C:
D:
E:
F:
G:
Command Window
fx >>
Workspace
Name
Value
Min
Max

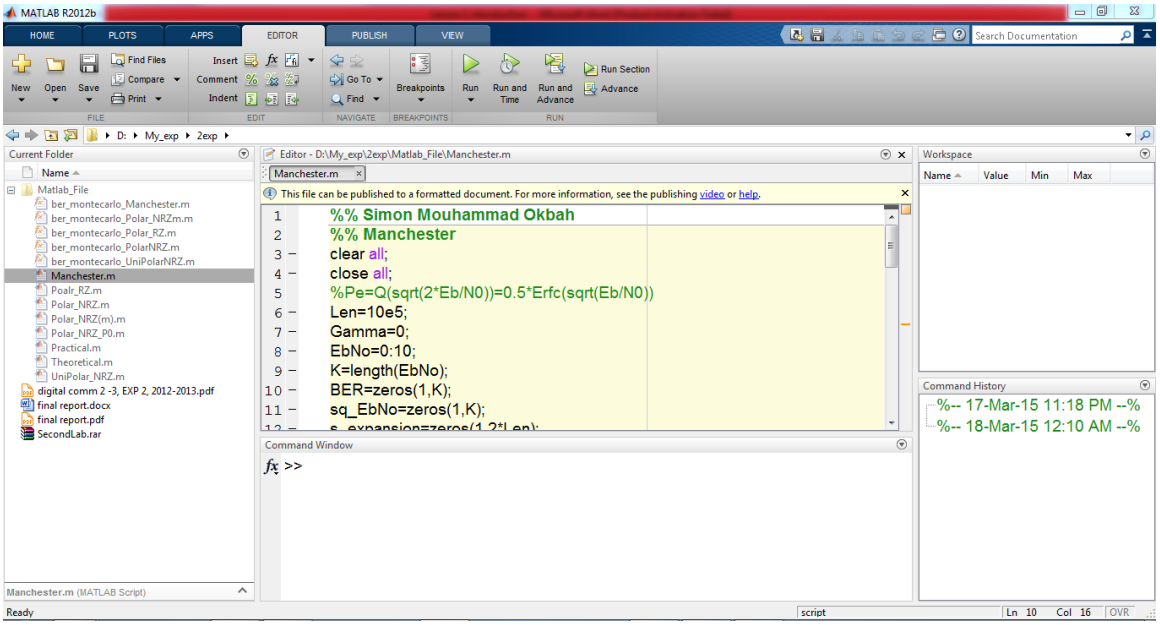

MATLAB R2012b
HOME
PLOTS
APPS
Search Documentation
New Script
New
Open
Find Files
Compare
Import Data
Save Workspace
New Variable
Open Variable
Clear Workspace
Analyze Code
Run and Time
Clear Commands
Simulink Library
Layout
Preferences
Set Path
Parallel
Help
Community
Request Support
FILE
VARIABLE
CODE
SIMULINK
ENVIRONMENT
RESOURCES
D: ▸
Current Folder
Name
3 rd year simon
4 th year simon
4th Project
5 th year simon
5th Project
Engineer
Master
My_exp
1exp
2exp
3exp
4exp
5exp
6exp
7exp
8exp
9exp
Other Projects
Program Files
References
SVU
Tutorials
Details
Command Window
fx >>
Workspace
Name
Value
Min
Max
Command History
%-- 17-Mar-15 11:18 PM --%
%-- 18-Mar-15 12:10 AM --%



الصورة التالية توضح الملفات الموجودة ضمن المجلد وهي ملفات توابع MATLAB لاحقتها m.

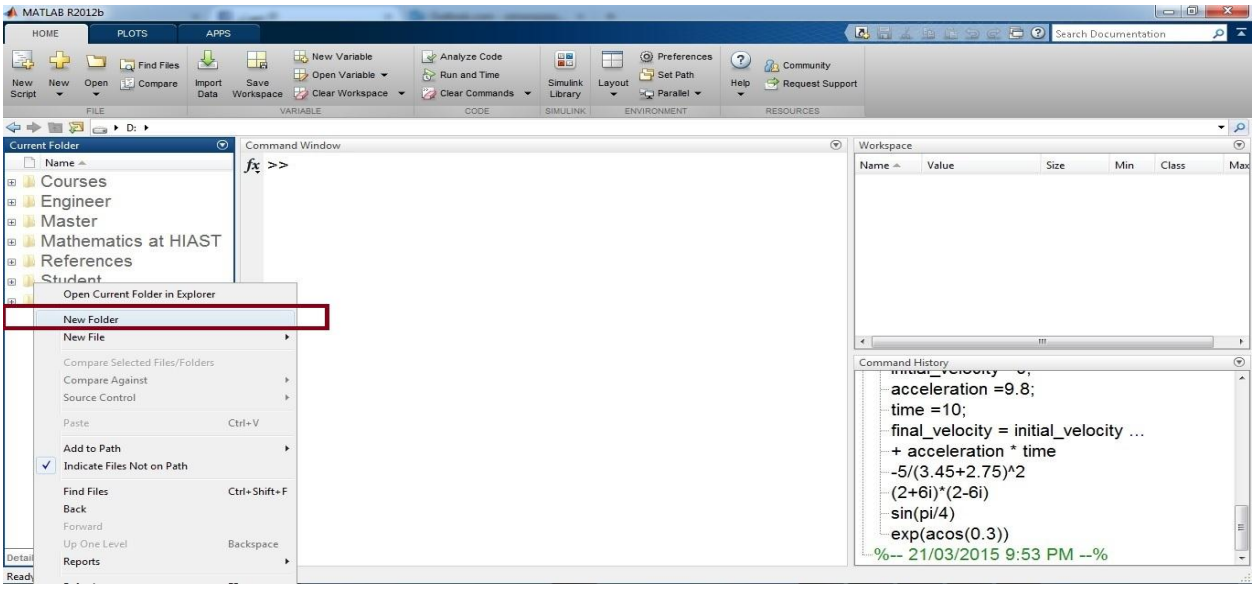


عند الضغط على الملف يتم فتحه



2. ينصح باستخدام المجلدات من أجل حفظ كافة البرامج والرمازات البرمجية بشكل منظم وسهولة الوصول إليها، فمثلاً إنشاء مجلد خاص بكل درس وتسمية الملفات بأسماء واضحة

يمكن ذلك ببساطة عبر الصورة التالية اضغط بالزر اليميني للفأرة ضمن current folder ستحصل على الخيارات التالية



ثم قم بإنشاء مجلد جديد

يمكن وضعها في نهاية الفصل تحت عنوان فيديوهات من المفيد الاطلاع عليها.

الروابط التالية من المفيد قرائتها

الأسئلة

1. كلمة MATLAB هي اختصار لأي كلمات ؟؟

مساعدة: قراءة فقرة التعرف على MATLAB.

2. مم يتألف نظام MATALB،

مساعدة: قراءة فقرة نظام MATLAB.

3. هل MATLAB لغة برمجة أم بيئة ؟؟

مساعدة: قراءة فقرة نظام MATLAB.

4. ما هو عنصر المعطيات الأساسي في برمجية MATLAB ؟؟ اختيار من متعدد

a. متحولات سلمية scalar

b. أشعة vectors

c. مصفوفات Matrices

d. صفيفات متعددة الأبعاد Multidimensional arrays

مساعدة: فقرة استخدامات MATLAB أو فكرة التمييز بين array و matrix التي طلب وضعها كفقرة او سؤال ضمن الفقرة السابقة.

5. ما هي أجزاء الواجهة الرئيسية من برمجية MATLAb؟

مساعدة: قراءة فقرة فهم الواجهة التخاطبية لبرمجية MATLAb.

6. كيف يتم الاستعلام عن عمل تعليمة ما ؟؟

مساعدة: Command window ضمن فقرة فهم الواجهة التخاطبية لبرمجية MATLAb.

الإجابات

1. هي اختصار ل Matrix Laboratory.

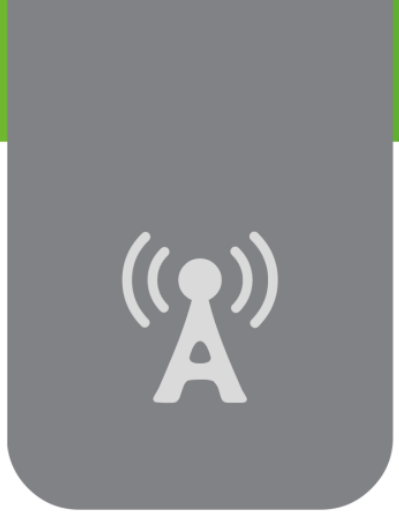
2. يتألف نظام MATLAB من
   a. لغة البرمجة MATLAB
   b. بيئة MATLAB
   c. Handle Graphics
   d. مكاتب التوابع الرياضية الخاصة ب MATLAB.

3. هي لغة برمجة وبيئة معاً
MATLAB هي لغة برمجة عالية المستوى قامت شركة MathWorks بتطوير برمجية وأسمتها MATLAB أيضاً حيث تتيح البرمجية MATLAB بتطوير برامج بلغة البرمجة MATLAB إضافةً لاستخدام أجزاء وأدوات أخرى مختلفة عن البرمجة بلغة MATLAB نذكر منها Simulink والأدوات Toolboxes سيتم التعرف عليهم لاحقاً.

4. الخيار رقم 4

5. هي خمسة أجزاء:
   a. Current folder
   b. Command Window
   c. Command History
   d. Workspace
   e. Toolstrip

6. نستطيع عن طريق Command Window كتابة Help ثم اسم التعليمة أو عن طريق Help browser.



الفصل الثاني: أساسيات لغة البرمجة MATLAB®
(النحو والمتحولات)
Fundamentals of MATLAB® Programming Language (Syntax and Variables)

عنوان الموضوع:

أساسيات لغة البرمجة MATLAB® (النحو والمتحولات)

Fundamentals of MATLAB® Programming Language (Syntax and Variables)

الكلمات المفتاحية:

MATLAB، MathWorks، workspace، command window، help، documentation، متحول variable، متحول سلمي scalar variable،شعاع vector، مصفوفة matrix، صفيفة array،نمط المعطيات data type، colon، مؤثرات operators، مؤثرات حسابية Arithmetic Operators، مؤثرات علاقاتية Relational Operators، مؤثرات منطقية Logical Operators، عمليات على المجموعات Set Operations، عمليات على مستوى البت Bit-Wise Operations.

ملخص:

نقدم في هذا الفصل أساسيات البرمجة باستخدام لغة MATLAB، يهتم الفصل بمحورين، الأول هو قواعد الكتابة ضمن اللغة (النحو Syntax) حيث سنتعرف ونستخدم بعض التعليمات لإجراء العديد من العمليات الحسابية، ثم يتم التعرف على المتحولات وكيفية التعامل معها ومعأشكالها المختلفة من متحول سلمي scalar، مروراً بالأشعة إلى المصفوفات ثنائية الأبعاد وانتهاءً بالصفيفة ثلاثية الأبعاد أو أكثر، وأخيراً نعرض المؤثرات ضمن MATLAB والتي تقسم إلى عدة أقسام منها المؤثرات الحسابية والعلاقاتية والمنطقية حيث سيتم عرض عمل كل مؤثر وأهميته إضافةً إلى مجموعة من الأمثلة لتوضيح الأفكار السابقة والاختلافات بين المؤثرات.

أهداف تعليمية:

يتعرف الطالب في هذا الفصل على:

- أساسيات لغة البرمجة MATLAB
- استخدام التعليمات، وقواعد الكتابة (النحو) Syntax.
- استخدام HELP الموجودة ضمن البرمجية
- تعريف المتحولات ضمن MATLAB
- التعامل مع متحولات سلميّة، أشعة، مصفوفات و صفيفات scalar variables, vectorsm matrices and arrays.
- التعامل مع المؤثرات الحسابية، العلاقاتية والمنطقية.

ماهو MATLAB وما هي استخداماته

النحو Syntax واستخدام التعليمات Commands

1. (Basic calculation) عمليات حسابية بسيطة

قبل البدء بعرض قدرات برمجية MATLAB الهائلة لابد لنا من معرفة أن هذه البرمجية يمكن أن تحل ببساطة مكان الآلة الحاسبةالبسيطة أو الآلة الحاسبة العلمية وحتى في مجال الأعداد العقدية، سنستعرض بعض العمليات البسيطة التي يمكن القيام بها ضمن command window من اجل التعرف على بعض قواعد الكتابة Syntax باستخدام لغة MATLAB.

بعد الرمز "»" ضمن command window يجري كتابة كافة التعليمات

مثلاً

- من أجل إجراء عملية الجمع اكتب 3+4 ثم اضغط enter النتيجة هي 7
- من أجل الرفع إلى قوة اكتب 3^5 ثم اضغط enter النتيجة هي 243
  - أخذ قيمة جيب أو تجيب زاوية، مثلاً $\sin\frac{\pi}{2}$ أو $\cos 0$ أو $\cos\frac{\pi}{2}$

اكتب sin(pi/2) ثم اضغط enter النتيجة هي 1

اكتب cos(0) ثم اضغط enter النتيجة هي 1

اكتب cos(pi/2) ثم اضغط enter النتيجة نظرياً هي 0 أما على الخرج هي 6.1232e-17 وهو عدد صغير جداً نتيجةً لتمثيل القيمة على عدد محدود من البتات وحسب نمط هذه القيمة

- التقسيم على صفر اكتب 4/0 ثم اضغط enter النتيجة هي لانهاية والخرج سيكون Inf للدلالة على infinity في بعض إصدارات البرمجية يظهر رسالة تحذير على الشكل " warning: division by zero"

يمكن إضافةً إلى العمليات السابقة القيام بالعديد من العمليات الحسابية كالقسمة والطرح مثلاً.

صور للخرج بعد تنفيذ العمليات ضمن Command Window


Command Window
>> 3+4
ans =
7
>> 3^5
ans =
243
>> sin (pi/2)
ans =
1
>> cos (0)
ans =
1
>> cos (pi/2)


Command Window
>> cos (0)
ans =
1
>> cos (pi/2)
ans =
6.1232e-17
>> 4/0
ans =
Inf
>> 4\0
ans =
0
>>

ملاحظة: خرج العمليات الحسابية يتم حفظه بتمحول اسمه ans ، وهو اختصار ل answer، وذلك عند تنفيذ أي عملية حسابية ضمن MATLAB والحصول على نتيجة وفي حال عدم إسناد الخرج إلى متحول عند إجراء عملية أخرى يجري إسناد ناتج العملية إلى المتحول

- مثلاً اكتب ضمن command window 6+8 نفذ هذه العملية ثم اكتب 4-9 ولاحظ التغييرات على قيمة ans.

2. محارف وقيم خاصة

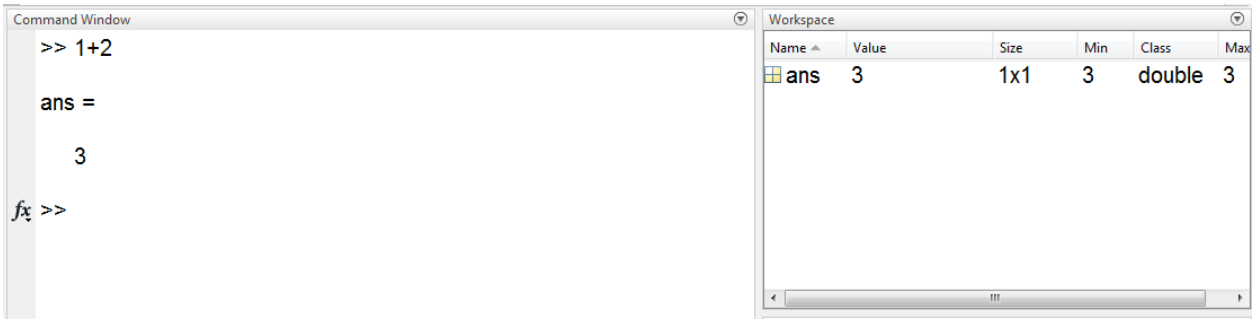
نعرض في هذه الفقرة بعض المحارف الخاصة باللغة والتي لها دلالاتها الخاصة بلغة MATLAB، كما نعرض بعض القيم Special Values الخاصة ضمن MATLAB أي التي لايمكن تسمية المتحولات بهذه الأسماء لأنها محجوزة ضمن اللغة الجدول التالي يبين هذه القيم والثوابت ودلالاتها:

Function	Return Value
ans	Most recent answer (variable). If you do not assign an output variable to an expression, MATLAB automatically stores the result in ans.
eps	Floating-point relative accuracy. This is the tolerance the MATLAB software uses in its calculations.
intmax	Largest 8-, 16-, 32-, or 64-bit integer your computer can represent.
intmin	Smallest 8-, 16-, 32-, or 64-bit integer your computer can represent.
realmax	Largest floating-point number your computer can represent.
realmin	Smallest positive floating-point number your computer can represent.
pi	3.1415926535897...
i, j	Imaginary unit.
inf	Infinity. Calculations like n/0, where n is any nonzero real value, result in inf.
NaN	Not a Number, an invalid numeric value. Expressions like 0/0 and inf/inf result in a NaN, as do arithmetic operations involving a NaN. Also, if n is complex with a zero real part, then n/0 returns a value with a NaN real part.
computer	Computer type.
version	MATLAB version string.

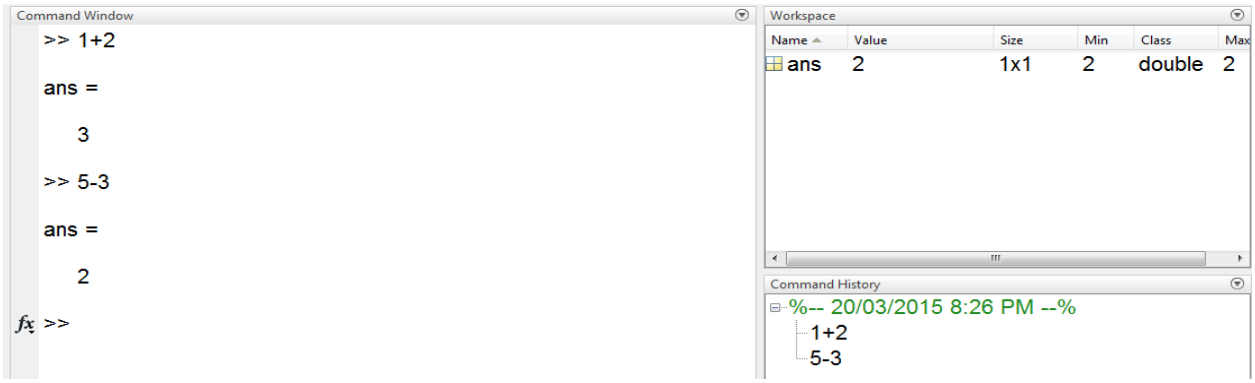
رابط يوجد ضمنه الجدول السابق

1- ans لعرض نتيجة أخر عملية حسابية تمت، وهو اختصار لكلمة answer، حيث MATLAB يقوم اتوماتيكياً بإسناد قيمة الخرج لأي عملية ضمن المتحول ans وذلك في حالة عدم إسناد قيمة الخرج من قبل المستخدم لمتحول ما. نعيد التذكير أنه في حال عدم إسناد الخرج إلى متحول عند إجراء عملية حسابية أخرى يجري أسناد ناتج العملية إلى المتحول ans.

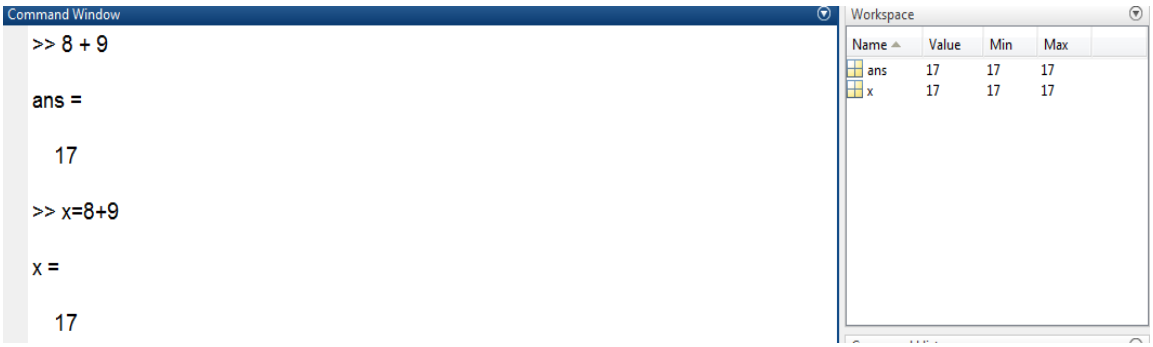
مثال اكتب 1+2 ضمن command window ثم اضغط enter نلاحظ ما يلي



قم بكتابة 3-5 ثم اضغط enter نلاحظ تغير قيمة ans



قم بعدها بتنفيذ العملية الحسابية 8+9 ثم عرف المتحول x على أنه خرج العملية أيضاً 8+9 نلاحظ أمه احتفظ بقيمة ans للعملية الأولى واحتفظ بقيمة العملية الثانية ضمن المتحول x، يمكن إجراء عملية مثلاً 4*5 وملاحظة تغيير قيمة ans.



2- Eps وهو اختصار ل epsilon أي عدد موجب تماماً وقيمته العددية صغيرة جداً وهي هنا تعبر عن الدقة النسبية عند تمثيل الأعداد باستخدام floating-point.

3- i أو j هو العدد التخيلي، ملاحظة في الإصدارات القديمة المن البرمجية يجري تعريف هذا العدد على الشكل التالي 1*i أو 1*j

4- Pi هو العدد المشهور π وقيمته 3.141592....

5- inf للدلالة على اللانهاية الموجبة، عند ظهور inf- يكون للدلالة على اللانهاية السالبة

6- NAN وهي اختصار ل Not-a-Number للدلالة على أنها ليست قيمة عددّية، وهي عبارة عن النتيجة من عمليات غير معرفة نتيجتها نذكر منها:

1. أي عملية حسابية على NAN
2. خرج عملية $\infty - \infty$ أو $\infty + \infty$ أو $0 * \infty$ أو $\frac{0}{0}$ أو $\frac{\infty}{\infty}$

تفيد في إنشاء مصفوفات بقيم ابتدائية غير صفرية وخاصةً في حالة مصفوفات من أحرف أو سلاسل محرفية

يمكن استخدام هذه القيم في الرمازات البرمجية مثلاً

```
r = 2 * pi
r = 6.2832
B = [1+2i 5-6i]
```

B = 1.0000 + 2.0000i 5.0000 - 6.0000i

وكذلك الأمر بالنسبة لباقي القيم ضمن الجدول السابق.

المحارف الخاصة Special Characters:

وهي: [ ] ( ) { } = ' . .. ... (). , : ; % %{ }% ! @

سنقوم بشرح بعض منها بشكل مختزل

[ ] تدعى Brackets وتستخدم من أجل إنشاء شعاع أو مصفوفة.

= المساواة من أجل عمليات الإسناد

' يدعى Matrix Transpose وهو يعطي المنقول والمرافق لمصفوفة ما (نفترض أن المصفوفة في الحالة العامة عقدية، في حال كانت حقيقة يعطي المنقول)

. لها عدة وظائف فنذكر منها تمثل الفاصلة العشرية، كما أنها المؤثر operator المستخدم في عمليات عنصر لعنصر element-by-element operation سيتم في هذا الفصل التعرف أكثر على التفاصيل عن هذا المؤثر الهام.

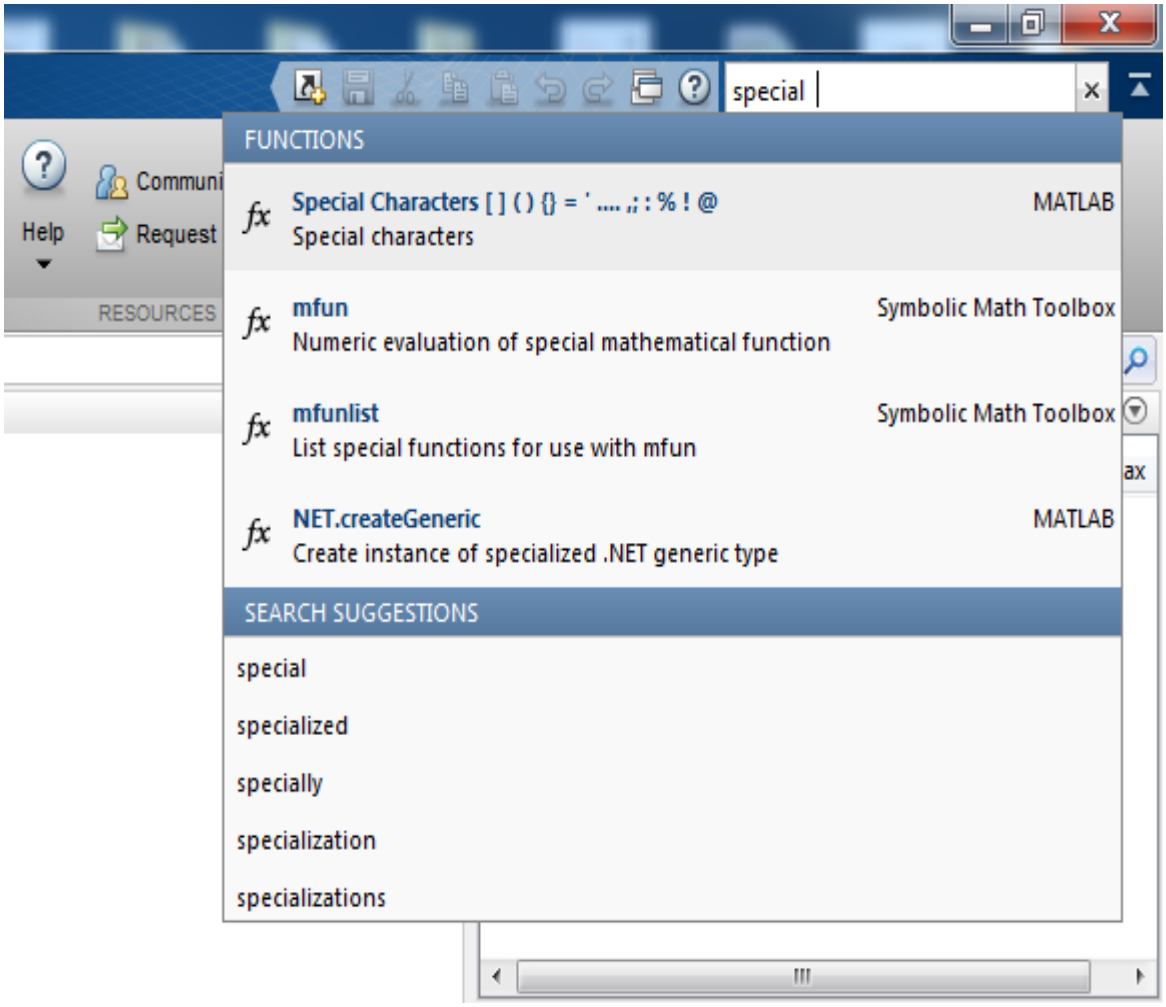
; تدعى semicolon تفيد في إنهاء السطر ضمن مصفوفة والانتقال لتعريف عمود جديد، كما أن وضعها بعد تعليمة ما يمنع نتيجة هذه التعليمة من الظهور ضمن Command Window.

, تدعى colon وتفيد في الفصل بين عناصر السطر الواحد ضمن شعاع او مصفوفة، كما انها تفيد في الفصل بين متحولات (محدّدات) التوابع function arguments.

: يدعى colon سيتم تقديم شرح مفصل عنه في فقرة العمل مع المتحولات ضمن MATLAB ضمن الفقرة الجزئية تعريف شعاع vector.

من أجل معرفة عمل باقي هذه المحارف ادخل إلى Help واكتب special characters وأيضاً symbol references كما هو موضح في الشكل ثم قم بقراءة الفقرات كاملةً.

ملاحظة هامة: سيتم ضمن الجلسة المتزامنة طرح العديد من الأسئلة عن هذه المحارف وعن عملها ضمن البرمجية، وسيتم التعامل مع هذه المحارف بشكل كبير والتآلف معها عند العمل مع البرمجية لكن من الضروري الاطلاع عليها.


special
FUNCTIONS
Special Characters [ ] ( ) {} = ' .... ,; : % ! @
Special characters
MATLAB
mfun
Numeric evaluation of special mathematical function
Symbolic Math Toolbox
mfunlist
List special functions for use with mfun
Symbolic Math Toolbox
NET.createGeneric
Create instance of specialized .NET generic type
MATLAB
SEARCH SUGGESTIONS
special
specialized
specially
specialization
specializations
Help
Request
RESOURCES

العمل مع المتحولات ضمن MATLAB

All MATLAB variables are multidimensional arrays, no matter what type of data. **A matrix is a two-dimensional array** often used for linear algebra

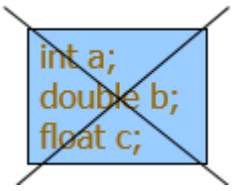
1. مقدمة عن المتحولات

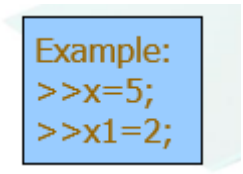
سيتم في هذه الفقرة تعلم كيف يتم إنشاء متحولات واستخدامها ولأخذ العلم فإن أي متحول ضمن MATLAB هو إما صفيفة متعدّدة الأبعاد Multidimensional Array، ويمكن أيضاً تعريف

- متحول سلميّ scalar ببعد 1×1
- شعاع vector ببعد $1 \times n$ او $n \times 1$ (يعبر عن سطر وحيد أو عمود وحيد)
- مصفوفة Matrix ثنائية الأبعاد ببعد $m \times n$

يمكن إنشاء ملف new script لتعريف المتحولات مما يفيد في حفظ هذا الملف واستخدامه لاحقاً (سيجري في الفصل القادم شرح مفصل عن ملفات scripts) أو يمكن تعريف المتحولات ضمن Command Window.

يعتبر إنشاء متحول جديد ضمن MATLAB هو أمر بسيط مقارنةً مع تعريف المتحولات ضمن مختلف لغات البرمجة مثل C، JAVA حيث لا توجد حاجة إلى تعريف نمط المتحول كـ Integer,Char,String, Double, Float, ... ويكفي فقط إسناد قيمة للمتحول وسيتم معرفة نمطه تلقائياً، إن غالبية المتحولات التي يجري التعامل معها هي من النمط double 64-bit و char 16-bit فهي الأنماط الافتراضية default

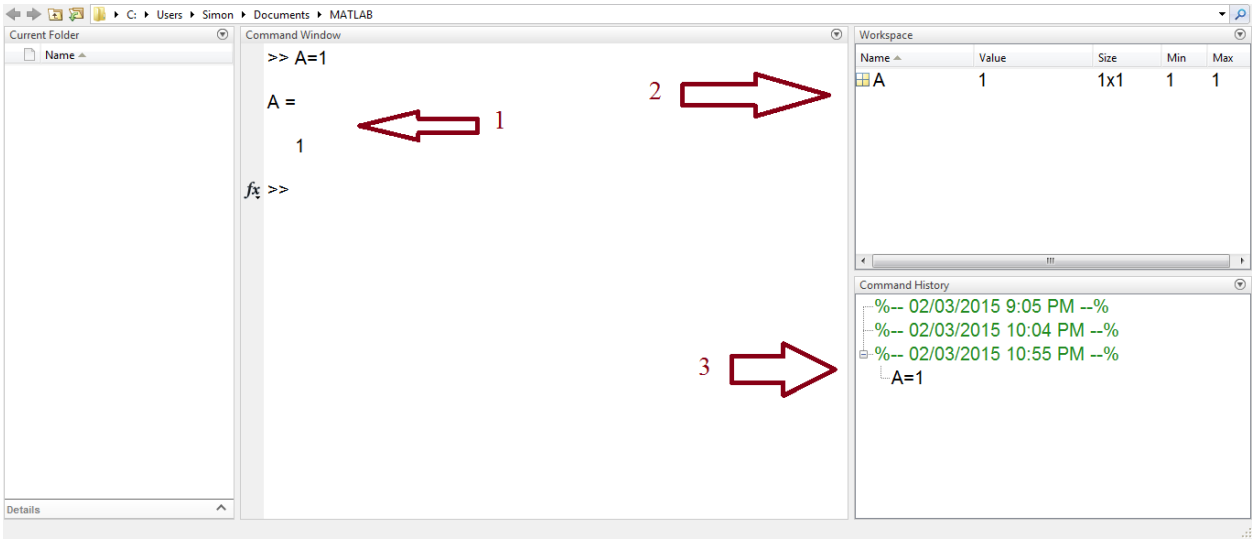




مثال:

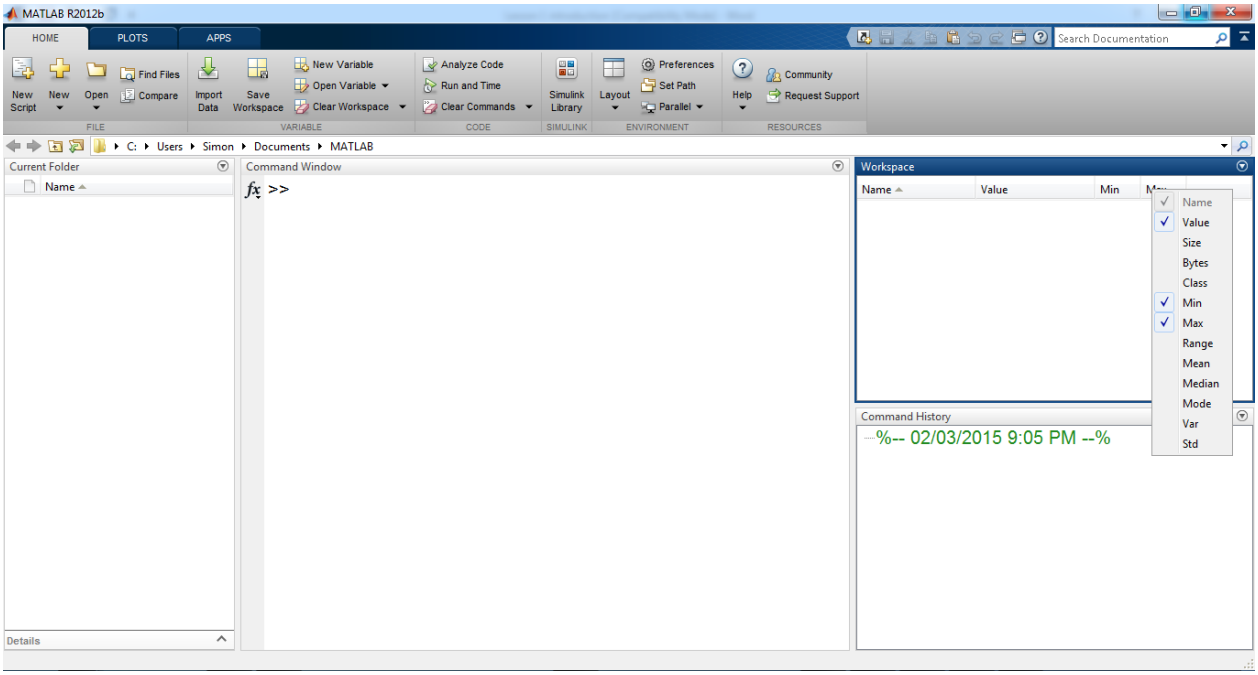
اكتب ضمن Command Window A=1 ثم enter

ستلاحظ التغيرات التالية ضمن واجهة البرمجية



نلاحظ أنه تم تعريف متحول باسم A على شكل مصفوفة بعدها 1 × 1 أي عبارة عن عنصر سلمي Scalar.

كما في الشكل التالي من أجل إظهار نمط هذا المتحول نقوم بالضغط على الزر اليميني على الشريط (Name, Value, ..., Max) ونختار إظهار Class. نلاحظ أن الصف Class هو Double



ملاحظة 1: الصف الافتراضي default Class لأي متحول هو double 64-bit ضمن برمجية MATLAB

سؤال كيف نقوم بتغيير الصف (أو نمط المتحول)؟؟

نقوم بما يلي ضمن Command Window نعرف

A=uint32(1) لتحويل المتحول إلى الصف unsigned integer 32 bit.

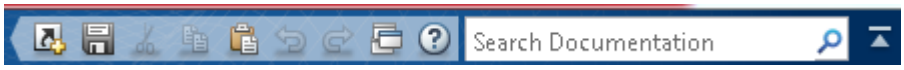
من المهم الاطلاع على هذه الأنماط ومعرفة دلالاتها لأنه سيتم استخدامها بشكل كبير في كامل المقرر)

سنقوم بتغيير الصف class للمتحول السابق إلى integer ونحدد عدد البتات الممكن تمثيل المتحول عليها وفقاً للصف المختار ثم سنوجد كافة الصفوف الممكن تعريفها باستخدام help

A=int16(1) من أجل جعل نمط المتحول integer ممثل على 16 bit

A=int8(1) من أجل جعل نمط المتحول integer ممثل على 8 bit

يبين الشكل غالبية أنماط المعطيات المستخدمة ضمن MATLAB يمكن الحصول على ذلك عن طريق كتابة Help class ضمن Command Window، او باستخدام Help الرئيسي ضمن البرمجية



اكتب ضمن Search Documentation الكلمة التالية Fundamental MATLAB Classes.

صورة 1

Data Type	Description
int8	8-bit signed integer
uint8	8-bit unsigned integer
int16	16-bit signed integer
uint16	16-bit unsigned integer
int32	32-bit signed integer
uint32	32-bit unsigned integer
int64	64-bit signed integer
uint64	64-bit unsigned integer

صورة 2

single	single precision numerical data
double	double precision numerical data
logical	logical values of 1 or 0, represent true and false respectively
char	character data (strings are stored as vector of characters)
cell array	array of indexed cells, each capable of storing an array of a different dimension and data type
structure	C-like structures, each structure having named fields capable of storing an array of a different dimension and data type
function handle	pointer to a function
user classes	objects constructed from a user-defined class
java classes	objects constructed from a Java class

ملاحظات حول تعريف المتحولات:

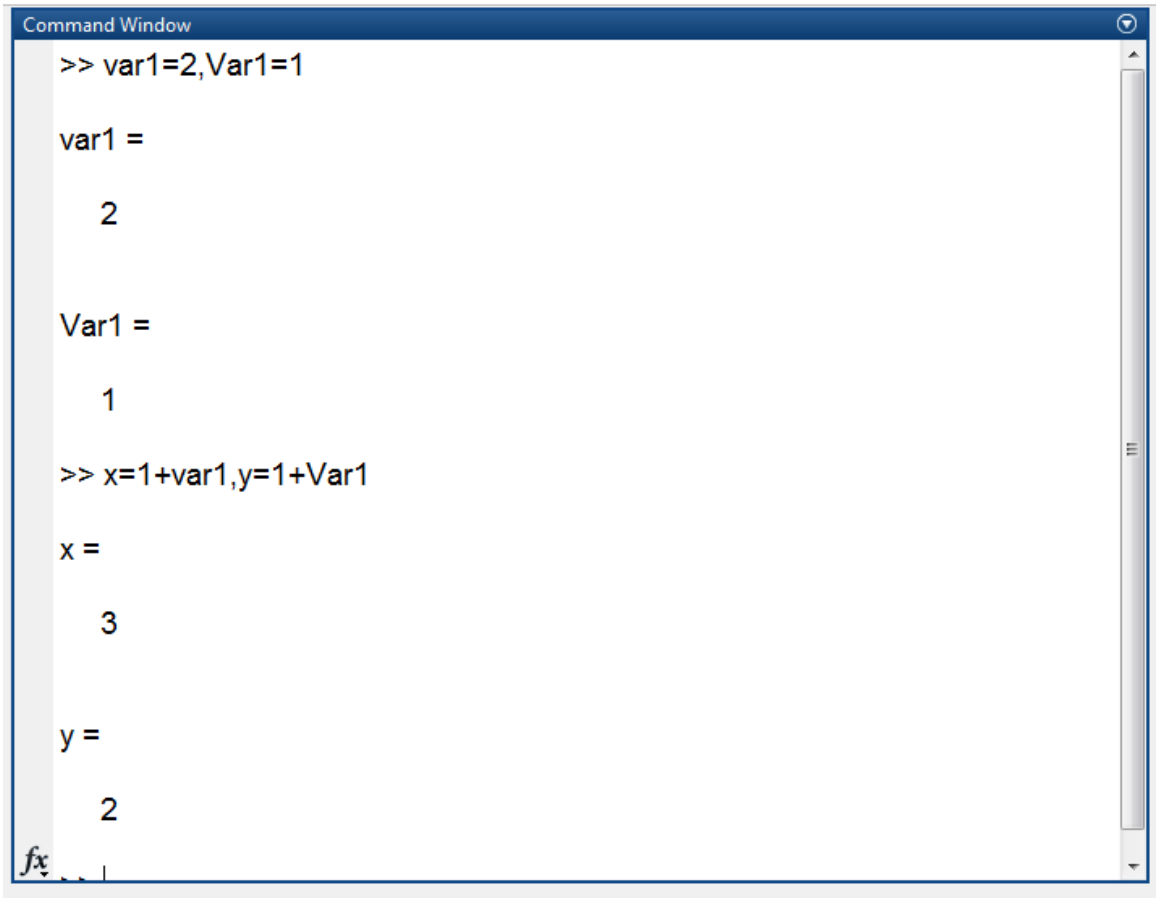
الرجاء الانتباه إلى ما يلي:

- يمكن الإشارة إلى المتحول أو استخدامه في عمليات لاحقة بعد تعريفه ضمن البرمجية.

- يجب أن تسند قيم إلى المتحولات قبل استخدامها.
- في حال عدم إسناد خرج أي عملية حسابية أو تعليمة أو تعبير expression، فإن البرمجية تقوم تلقائياً بإسناده للمتحول ans حيث يمكن استخدامه لاحقاً ولكن عند تنفيذ أي تعليمة أو عملية حسابية أخرى يتم إسناد آخر نتيجة للمتحول ans.
- حول أسماء المتحولات
  - اول حرف من الاسم يجب ان يكون حرف LETTER
  - بعد ذلك يسمح بأي تشكيلة من الحروف والأرقام و _ (underscore)
  - Case Sensitive أي ان المتحول var1 مختلف عن Var1
  - لا يمكن استخدام الكلمات المفتاحية أو الكلمات المحجوزة ضمن اللغة مثل if, else, for, while,....

مثال من اجل اسماء المتحولات للتوضيح

قم بتعريف المتحولات التالية var1=2, Var1=1 ضمن Command Window ثم عرّف المتحولين التاليين x=1+var1, y=1+Var1 ثم نفذ التعليمات ولاحظ الخرج



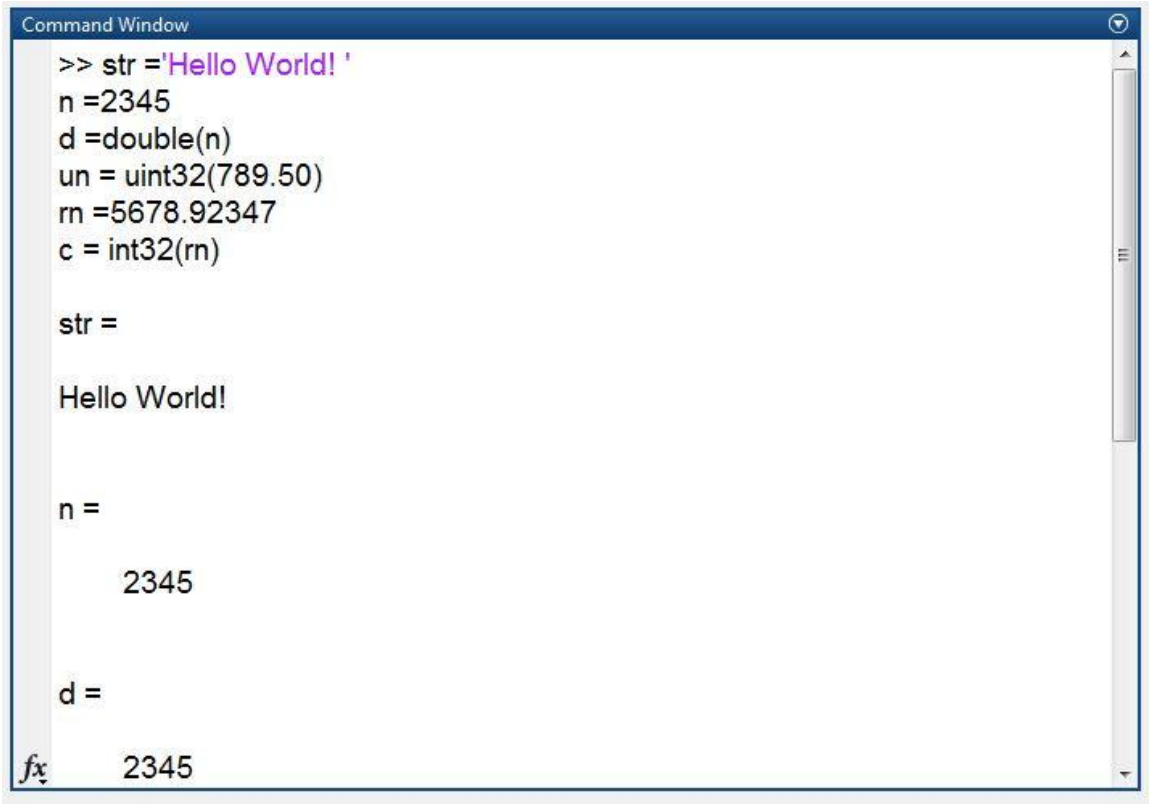
مثال: قم بتعريف المتحولات التالية:

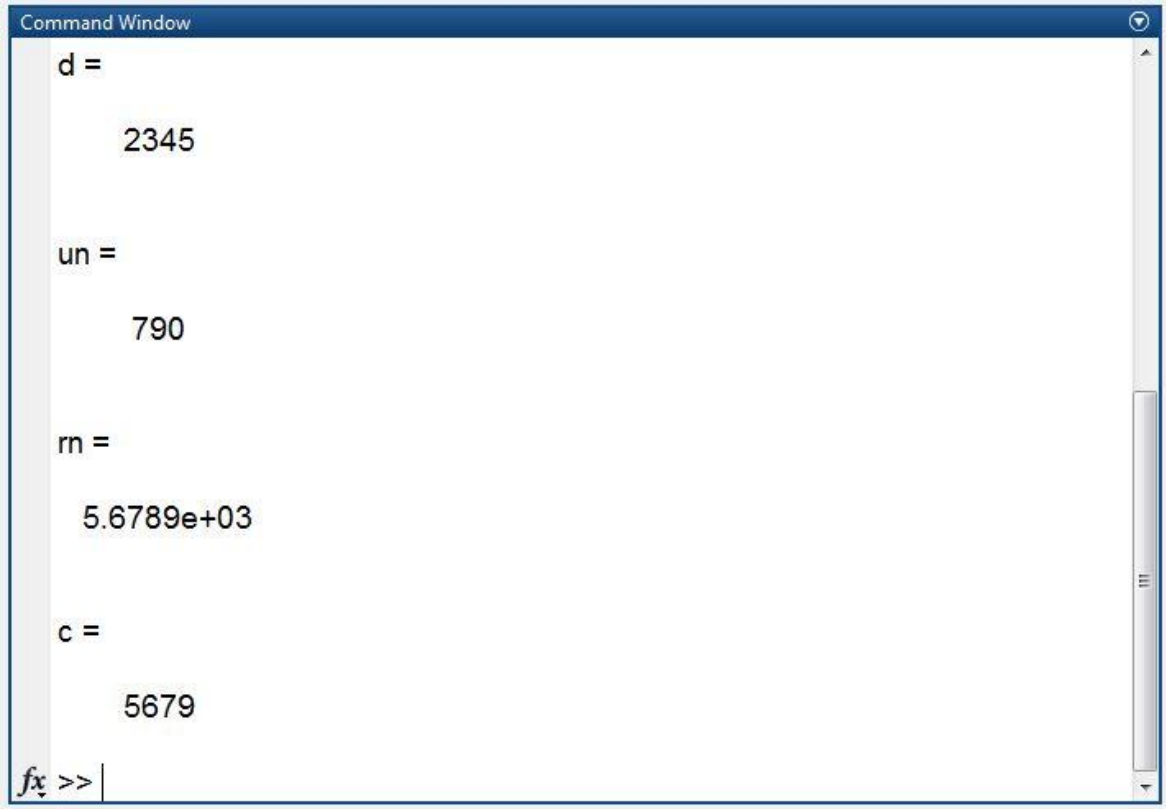
str ='Hello World! '

n =2345

```
d =double(n)
un = uint32(789.50)
rn =5678.92347
c = int32(rn)
```

الخرج هو:





نذكر بالتعليمات التالية:

clc من أجل مسح محتوى Command Window فقط أي مسح التعليمات التي قمنا بإدخالها ومن ضمنها تعريف المتحولات والنتائج التي حصلنا عليها، مع التنبيه إلى أن المتحولات والنتائج يبقوا معرفين ضمن workspace أي عمل التعليمة هو فقط مسح شاشة الإدخال والخرج.

أما من أجل مسح متحول ما نقوم بكتابة clear ثم يليها اسم المتحول الذي نرغب بمسحه أي

clear nameofvariable

إزا اردنا مسح المتحول c نكتب clear c، وإزا أردنا مسح جميع المتحولات نكتب التعليمة clear all.

قم بتجريب التعليمات السابقة على المتحولات التي جرى تعريفها سابقاً، أولاً جرّب clc ثم clear c ثم clear all.

2. كيف نعرف متحول سلمي scalar

ننوه منذ الآن وإلى نهاية المقرر إلى ضرورة تعريف المتحولات بأسماء تعبر عن استخدامها على سبيل المثال، متحول يعبر عن مطال الإشارة يسمى مثلاً Amp كاختصار لكلمة Amplitude، متحول عبر عن تردّد يسمى مثلاً Freq وهكذا.

لتعريف متحول سلمي scalar نقوم بتعريف اسمه وإسناد قيمته وتلقائياً يتم تعريف نمطه (أو الصف)على أنه double 64-bit

x=125;

إن إضافة % يسمح لنا بإضافة تعليق أي عند تنفيذ التعليمات لا يتم تنفيذ ما هو مكتوب بعد الإشارة % تفيد التعليقات في وضع شرح عن الرماز البرمجي المكتوب أو توضيح عن المتحولات ودلالاتها، وهذا يفيد عند العودة للرمازات البرمجية لاحقاً أو عند تسليم عمل على شكل مشروع أو وظيفة، وننوه إلى ضرورة استخدام التعليقات لتوضيح مهمة كل متحول أو توضيح خطوات العمل عند برمجة خوارزمية مثلاً.

يمكن منع إظهار الخرج عن طريق إضافة " ; " بعد التعليمة أو المتحول كما في المثال التالي تفيد هذه التعليمة كثيراً في حالة الرمازات البرمجية الكبيرة وتنجيز الخوارزميات حيث الاهتمام فقط بالخرج النهائي وليس بخرج كل سطر أي من أجل منع خرج كل تعليمة من الظهور ضمن Command Window يجب استخدام " ; " مجموعة من الأمثلة:

- عرف المتحولات التالية ضمن Command Window

x= 125; % scalar variable with value of 125

Freq= 8000; % Sampling Frequency

T=10; % period =10 Sec

A=1; % amplitude =1 Volts

y =3; % defining y and initializing it with a value

سيقوم MATLAB بتنفيذ التعليمات السابقة وتعريف المتحولات وسيتم إنشاء مصفوفة ببعد 1×1 وكل منها باسم معرّف باسم المتحول ويخزن ضمن كل من المتحولات السابقة القيمة المسندة له.

- يمكن أيضاً تعريف متحول هو خرج لتعليمة أو عملية أو تابع ما

مثلاً:

Z=sqrt(25)

- كما يمكن إجراء عدة إسنادات للمتحولات في نفس السطر

مثلاً اكتب

a=1; b=2; c=a*b;

ملاحظة: يمكن الاستفادة من التعليمة who من أجل عرض كل أسماء المتحولات التي جرى تعريفها سابقاً.

تمرين

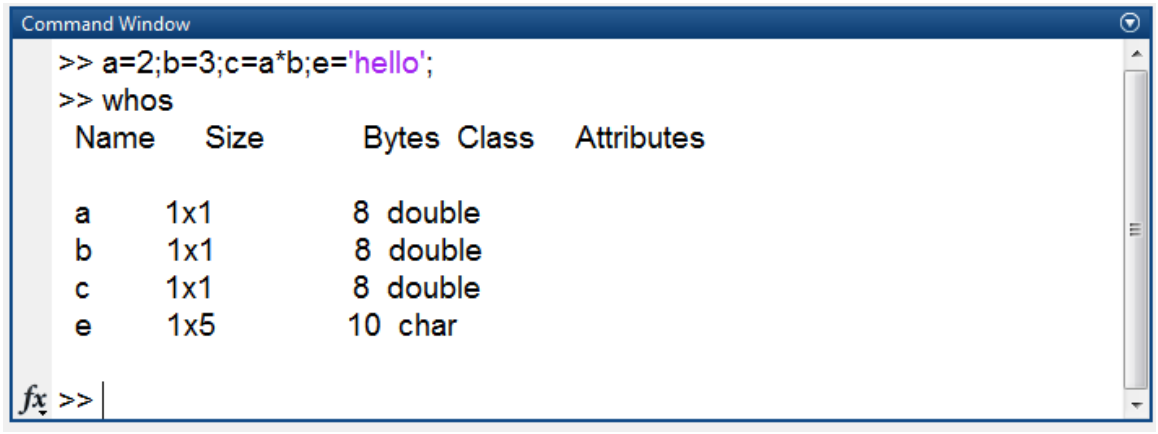
استخدم help لمعرفة فائدة التعليمة whos ثم قم بتعريف المتحولات a=2;b=3;c=a*b;e='hello'; ونفذ بعدها التعليمة whos

الحل

تفيد التعليمة whos في معرفة

whos is a long form of WHO. It lists all the variables in the current workspace, together with information about their size, bytes, class, etc.

عند التنفيذ نحصل على



انتهى المثال

ملاحظة: عند تعريف بعض المعادلات أو العلاقات الجبرية قد تكون طويلة في الصياغة، وخاصةً عند استخدام أسماء طويلة للمتحولات، لذلك يفيد المؤثر "..." في حالة long assignment عند استخدامه

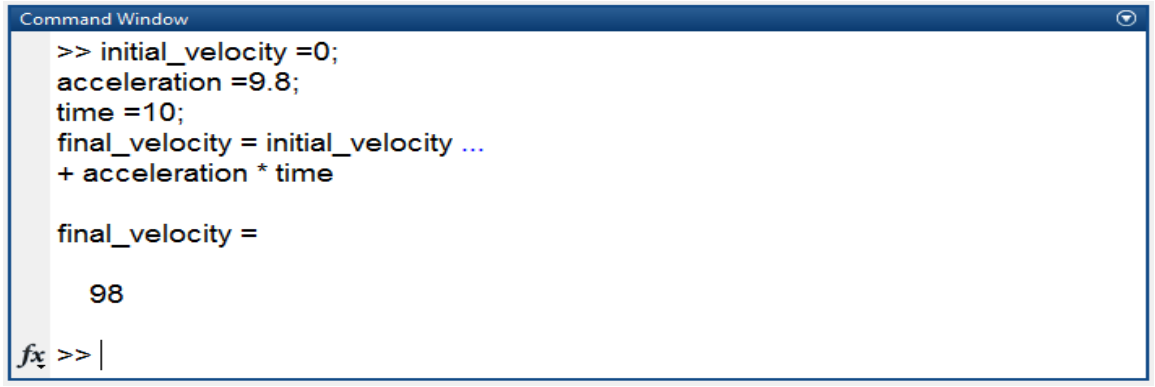
مثال للتوضيح للملاحظة السابقة

عرف المتحولات التالية مع القيم البدائية واستخدم ... للانتقال إلى سطر جديد ضمن تعريف المتحول الأخير

حيث ان العلاقة هي final_velocity=initial_velocity+acceleration*time

```
initial_velocity =0;
acceleration =9.8;
time =10;
final_velocity = initial_velocity...
 +acceleration * time
```

خرج MATLAB بعد التنفيذ



يتم عرض النتائج ضمن MATLAB باستخدام النمط الافتراضي default وهو double 64-bit إلا أن شكل العرض format الافتراضي هو short format أي يتم عرض أربعة أرقام عشرية بعد الفاصلة، من أجل زيادة الدقة precision (أي عدد الأرقام التي يتم عرضها بعد الفاصلة في ناتج أي عملية حسابية وعلى اعتبار أن نمط التمثيل هو double 64-bit) يمكن استخدام long format لنوضح الاختلاف لدينا المثال التالي

اكتب ضمن Command Window

```
format long
x =7+10/3+5^1.2
```

ولاحظ الخرج

ثم اكتب ضمن Command Window

```
format short
x =7+10/3+5^1.2
```

ولاحظ الخرج نجد أن

```
x =
17.231981640639408
x =
17.2320
```

The format long command displays 16 digits after decimal

The format short command displays 4 digits after decimal

تتمة للمثال

من أجل معرفة أكثر عن الأنماط Style المتوفرة ضمن التعليمة format

help format ثم ابحث ضمن Input Arguments ستجد style قم بقراءة وتجريب الأنماط المختلفة.

كما يمكن التعامل مع متحولات على انها string أو char

لتعريف النمط Char للمتحول يتم على الشكل

```
B= char ('s')
```

أما لتعريف string يمكن بالأشكال التالية:

```
S= 'Any Characters'
S= [S1 S2 ...]
```

يوجد أشكال أخرى لتعريف string يجب الاطلاع عليها عن طريق help string

يمكن الاستفادة أيضاً من التعليمات التالية num2str و str2num من أجل التحويل من عدد إلى محرف والعكس، باستخدام help تعلم كيفية استخدام التعليمة

ويمكن أيضاً دمج سلستين محرفيتين عن طريق التعليمة strcat(اطلع على help لمعرفة المزيد عن هذه التعليمة) مثال

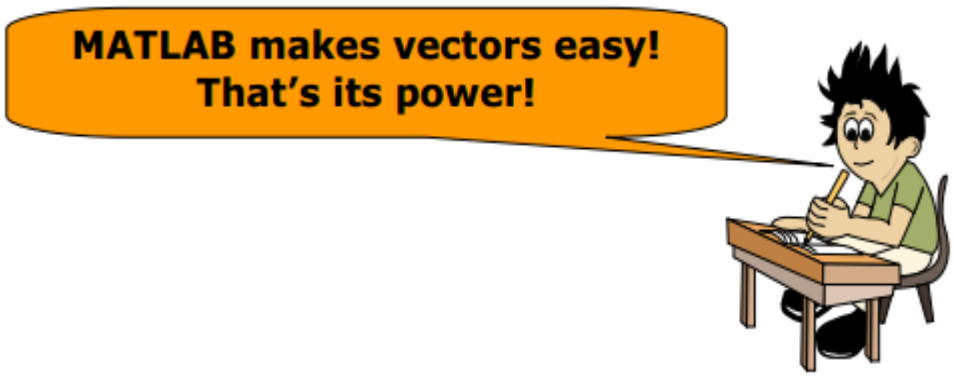
قم بتعريف المتحولات التالية

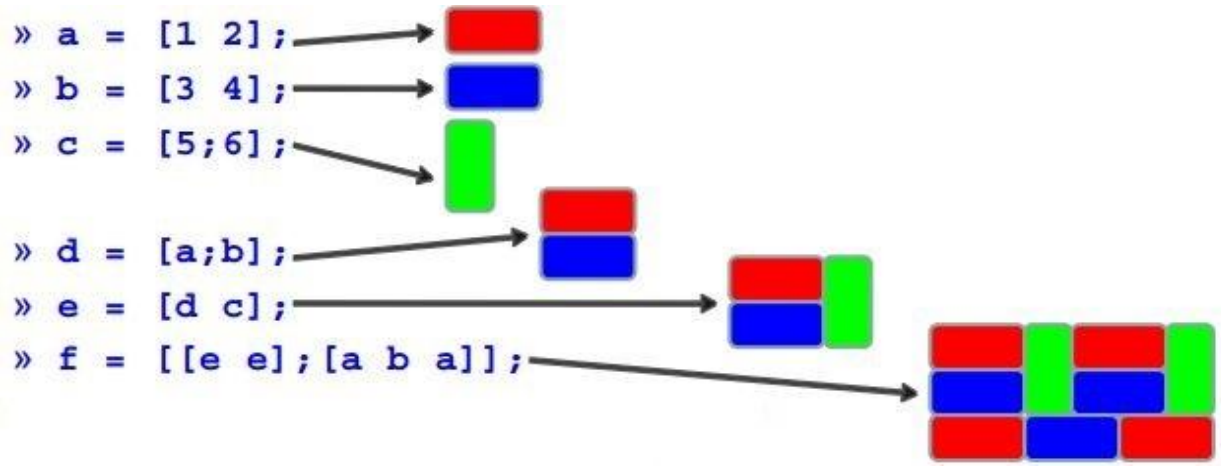
```
B='Hello'
C='World'
D=strcat(B,C)
```

الخرج هو

```
Command Window
>> B='Hello'
C='World'
D=strcat(B,C)

B =

Hello


C =

World


D =

HelloWorld

fx >>
```

3. كيف نعرف شعاع Vector





يمكن تعريف شعاع vector على الشكل

V= [1, 2, 3] or V1 = [1 2 3]

ملاحظة: للفصل بين عناصر الشعاع ضمن نفس السطر نستخدم " " أو ","

V= [1 2 3 4 5];

نحصل على شعاع ببعد 1×5

V1= [1,7,123,66,12];

نحصل على شعاع ببعد 5×1. أما من أجل الحصول على على شعاع ببعد 1×5 فيكون على الشكل التالي

V2= [1
2
44
41
]56

أو على الشكل v=[1;2;3;4;5;6]

ملاحظة: من أجل الفصل بين أسطر الشعاع نستخدم ";".

لأخذ مرافق ومنقول transpose لشعاع أو مصوفة نستخدم " ' "، قم بتجريب ذلك على الشعاع V1، V2.

من أجل أخذ المرافق والمنقول معاً، يجب توليد قيمة عقدية أولاً، مثلاً ;x=1+ 5 *i

ثم نفذ y=x'.

تدريب: ولد شعاع عقدي مؤلف من 5 قيم لا على التعيين وقم بإيجاد مرافق ومنقول هذا الشعاع

الحل

V_complex= [2+2*i, 3*i, 1-10*i, 77, 3+10*i]

V_tr=V_complex'

الشكل العام لتوليد شعاع

من أجل توليد شعاع على الشكل vect=[1,2,3, ….,n] أي ببعد $1 \times n$ يتم ذلك بالشكل التالي

vect1= start: step: stop

وهنا نقوم بتعريف المتحولات السابقة (start,step, stop) حسب الشعاع المطلوب، إن طول الشعاع المولّد هو

$$length = \frac{stop - start}{step} + step$$

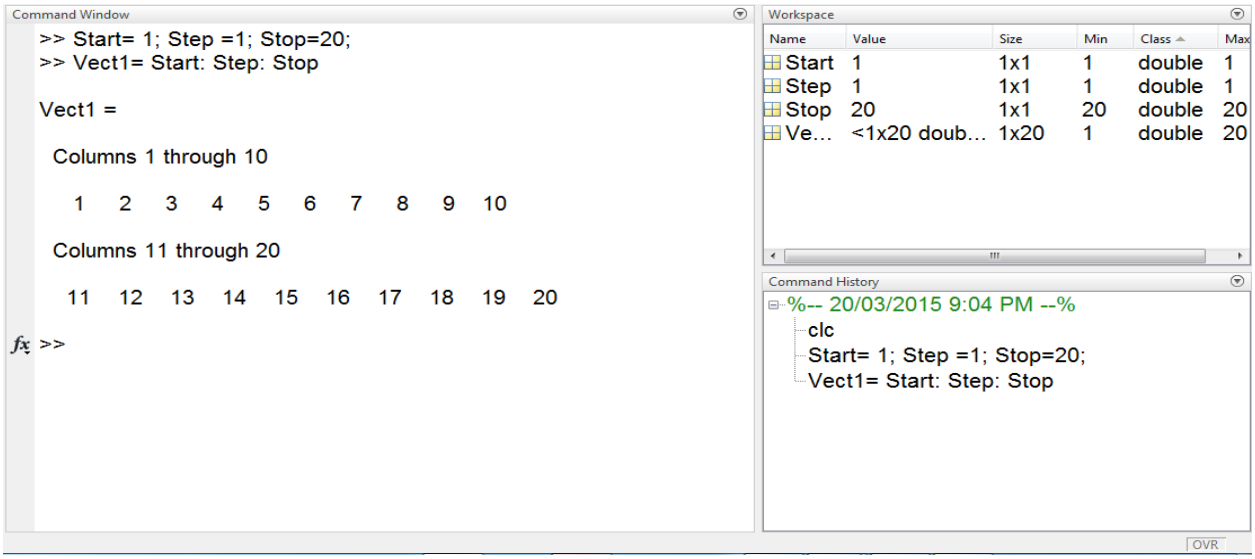
مثلاً لتوليد الشعاع [1,2, …, 20] =vect نقوم بمايلي

نعرف المتحولات التالية

Start= 1; Step =1; Stop=20;

ثم نعرف الشعاع على الشكل التالي

Vect1= Start: Step: Stop



ملاحظة إن تعريف الشعاع بالشكل التالي

Vect=start: stop

يعطي نفس النتيجة لأن الخطوة الافتراضية في برنامج هي 1.

من أجل معرفة طول شعاع يمكن استخدام التعليمة length وهي تستخدم على الشكل التالي:

a = [1 2 3 4];

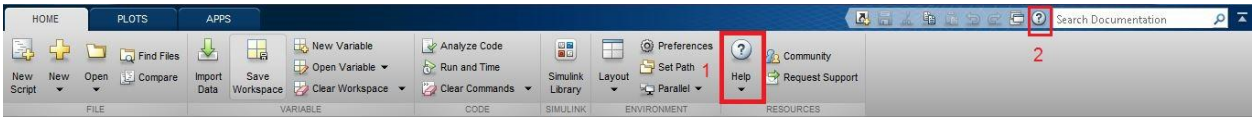
length (a)

الإجابة هي 4.

للاستعلام أكثر عن التعليمة length استخدم MATLAB Help وفق تسلسل البحث التالي:

ادخل إلى HELP ضمن toolstrip جزء home قسم resources واختر Documentation هو رقم 1

بالصورة أو اضغط على الاختصار هو رقم 2 بالصورة



ادخل إلى MATLAB ثم Language Fundamentals ثم Matrices and Arrays ثم Array Dimensions وادخل إلى التابع length.

ملاحظة: مطلوب من الطلاب البحث ضمن Help عن التعليمات التالية وفهم عملها:

linspace و logspace.

4. كيف نعرف مصفوفة Matrix

يتم التعامل ضمن MATLAB على أساس المصفوفات، وهذا هو الاختلاف الجوهري بين MATLAB وباقي لغات البرمجة، حيث تم بناء برمجية MATLAB للتعامل مع المصفوفات بمختلف الأبعاد والحجوم، مما يجعلها من أهم البرمجيات في مجال التحليل الإحصائي Statistical Analysis.

لتعريف مصفوفة 3 × 2 نقوم بما يلي

```
M= [1 2 3; 4 5 6]
```

تعريف مصفوفة 4 × 2

```
M0= [11,4,0,2
     4,5,8,9]
```

لتعريف مصفوفة 2 × 2 يمكن استخدام التعريفات التالية وهي متكافئة

```
M1= [1 2; 3 4]
M2= [1,2;3,4]
M3 = [1 2
      3 4 ]
```

للفصل بين عناصر المصفوفة ضمن نفس السطر نستخدم " " أو "," أما من أجل الفصل بين الأسطر نستخدم ";".

يوجد العديد من التوابع المفيدة للتعامل مع المصفوفات نذكر منها:

size تفيد في معرفة حجم صفيفة، حيث تقوم هذه التعليمة بإرجاع متحول هو شعاع كل عنصر فيه هو أحد أبعاد الصفيفة أي في حال إنشاء مصفوفة ثنائية البعد فالخرج هو شعاع يحتوي عنصرين الأول هو عدد الأسطر والثاني هو عدد الأعمدة.

sum يفيد هذا التابع في معرفة مجموع عناصر صفيفة حيث يقوم بجمع عناصر كل عمود.

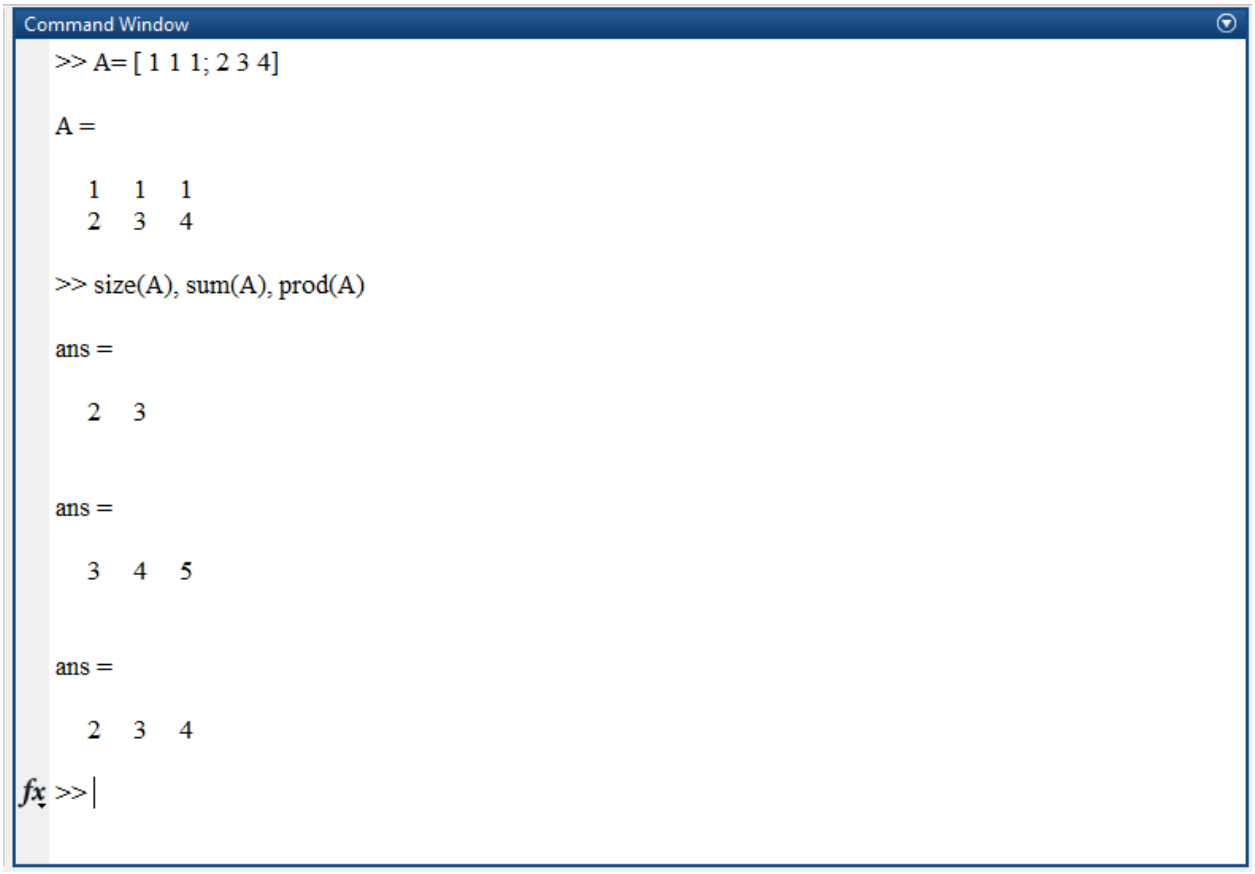
prod يفيد هذا التابع في معرفة حاصل ضرب عناصر صفيفة حيث يقوم بضرب عناصر كل عمود.

مثال: لنقم بإنشاء المصفوفة التالية A= [1 1 1; 2 3 4] ولنطبق التعليمات السابقة على الترتيب أي size(A), sum(A), prod(A).

ملاحظة: يمكن التعرف أكثر على هذه التعليمات عن طريق Help، كما يمكن إسناد خرج هذه التعليمات إلى متحولات بغرض الاستفادة منها في تطبيق ما أو ضمن العمل على موضوع ما.

ملاحظة: يمكن لتابع ... ,tan , sin, cos أن تاخذ دخل هو مصفوفة، جرّب ذلك عن طريق توليد مصفوفة سمّها MAT ثم أخذ sin(MAT) و كذلك الأمر للتعليمات السابقة. ويوجد أيضاً عدد من التعليمات المفيدة اطلّع على التعليمة sort.

ملاحظة: يوجد عدد كبير من التعليمات والتوابع التي تسهل عمل المستخدم مع المصفوفات يمكن الاطلاع عليها عن طريق help وفهم معناها، قمنا بعرض عدد قليل وقليل جداً من التعليمات ونترك للمستخدم التعلم عند الحاجة.



5. كيف نعرف صفيفة Array

قمنا في الفقرات السابقة بشرح الطرائق المختلفة لتمثيل المعطيات ضمن MATLAB إما على شكل متحول سلمي أو شعاع أو مصفوفة أي كصفيفة array أحادية الأبعاد أو ثنائية الأبعاد، إلا أن MATLAB يتيح لنا إنشاء أبعاد بأبعاد أكثر مثلاً صفيفة ثلاثية الأبعاد ويتم ذلك بسهولة ولكن المشكلة تكمن في عرض الصفيفة لأنه من غير الممكن عرض أشكال ثلاثية الأبعاد على شاشة ثنائية الأبعاد، لذلك نقوم بالتعامل مع مساقط هذه المصفوفة.

ببساطة يتم تعريف الصفيفة عن طريق تعريف الأبعاد وقيم عناصرها.

قم باستخدام help بالبحث عن التعليمات التالية zeros, ones, rand

بعد البحث يجب معرفة أن

zeros تقوم بإنشاء صفيفة كل قيمها أصفار

ones تقوم بإنشاء صفيفة كل قيمها واحدات

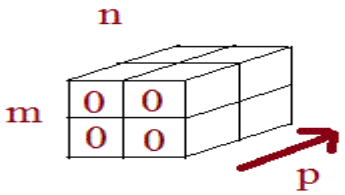
rand تقوم بإنشاء صفيفة بتوزيع شبه عشوائي هو توزيع منتظم على المجال [0,1]

بعد التعرف على التعليمات **المهمة** السابقة نقوم بتعريف صفيفة ثلاثية الأبعاد على الشكل

نكتب zeros(2,2,2) ثم enter

يتم التعريف على الشكل التالي كما يتم عرض مساقط هذه الصفيفة

```
Command Window
>> zeros(2,2,2)

ans(:,:,1) =

     0     0
     0     0


ans(:,:,2) =

     0     0
     0     0
```



هذه الصورة هي لتوضيح الصفيفة <----------

حيث m هو عدد الأسطر، n هو عدد الأعمدة و p هو البعد الثالث

إن المؤثر Operator ":" في أسطر وأعمدة الصفيفة السابقة

يدل على أنه يتم إظهار جميع العناصر الموجودة في هذا البعد اي هنا يتم عرض عناصر البعدين الأول والثاني من أجل العمود الأول أي p=1 ثم يتم عرض عناصر البعدين الأول والثاني من أجل العمود الثاني أي p=2 اعتماداً على الشكل السابق

جرّب command window ما يلي ضمن ans(:,1,:) او ans(:,2,:) سينتج الشكل التالي

```
>> ans(:,1,:)

ans(:,:,1) =

     0
     0


ans(:,:,2) =

     0
     0
```

- يمكن أيضاً اعتماداً على تعريف كل مسقط للصفيفة تعريف صفيفات بأبعاد مختلفة أكبر من بعدين
- ويمكن تعريف أيضاً مصفوفة مصفوفات على الشكل التالي:

تمرين: قم بتعريف المصفوفات التالية

a=[1,2;3,4] b=[5,6;7,8] c=[9,9] d=[4;4;4]

من أجل إنشاء صفيفة تحتوي كافة المصفوفات السابقة نقوم بتعريف

e={a,b;c,d}

فنحصل على

```
e =

    [2x2 double]    [2x2 double]
    [1x2 double]    [3x1 double]
```

من أجل استعمال (أو معاينة) عناصر هذه المصفوفة التي هي عبارة عن مصفوفات يمكن كتابة e{1,1} والخرج هو عبارة عن المصفوفة a.

```
>> e{1,1}

ans =

     1     2
     3     4
```

Operators in MATLAB المؤثرات ضمن MATLAB

أولاً سنقوم بشرح المؤثر Colon ثم سننتقل لشرح باقي المؤثرات.

المؤثر Colon(:) ويعتبر من أهم المؤثرات في MATLAB، يستخدم في توليد سلسلة من الأرقام وهي تفيد في إنشاء (توليد) وفهرسة (indexing) الأشعة والمصفوفات و الصفيفة array، كما أنه يستخدم ضمن حلقة for من أجل إنشاء عداد الحلقة.

سنقوم بشرح استخدامات هذا المؤثر:

- إنشاء شعاع بخطوة ثابتة أو متغيرة كما أوضحنا سابقاً في هذه الفقرة

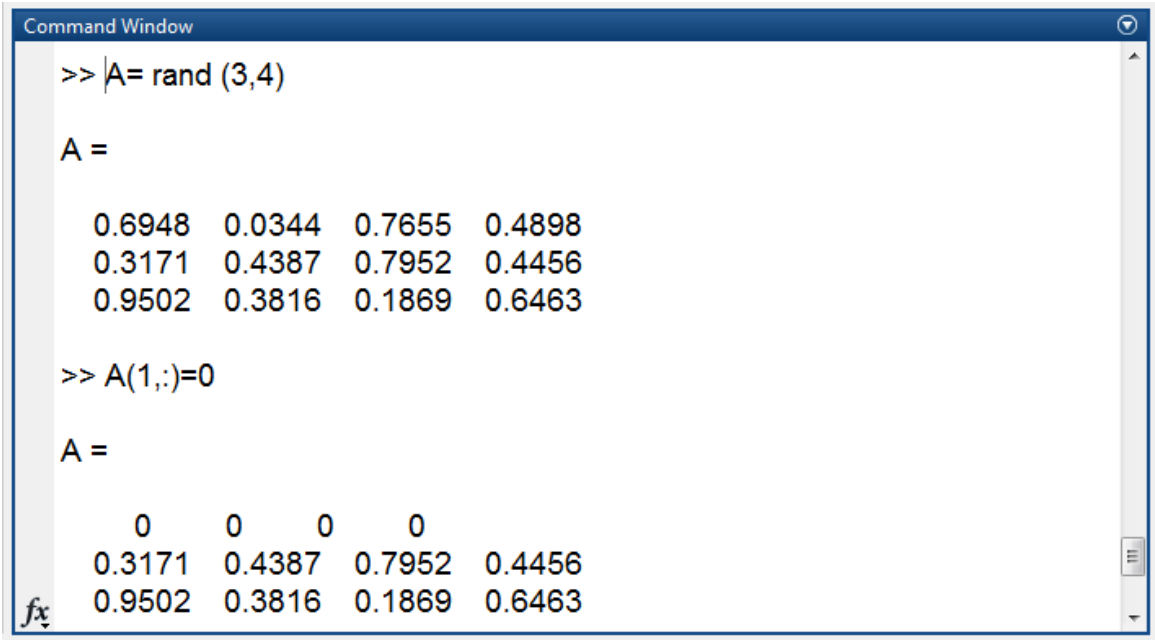
```
Vect = 1:15;
Vect_2 = 0:2:30;
```

- تفيد في إسناد قيم معينة إلى مصفوفة ما بعد تعريفها أو إسناد قيم جديدة للمصفوفة دون تغيير شكلها.

مثال: عرّف المصفوفة التالية A= rand(3,4) ثم قم بما يلي:

```
A(1,:) = 0
A(:) = 1:12
```

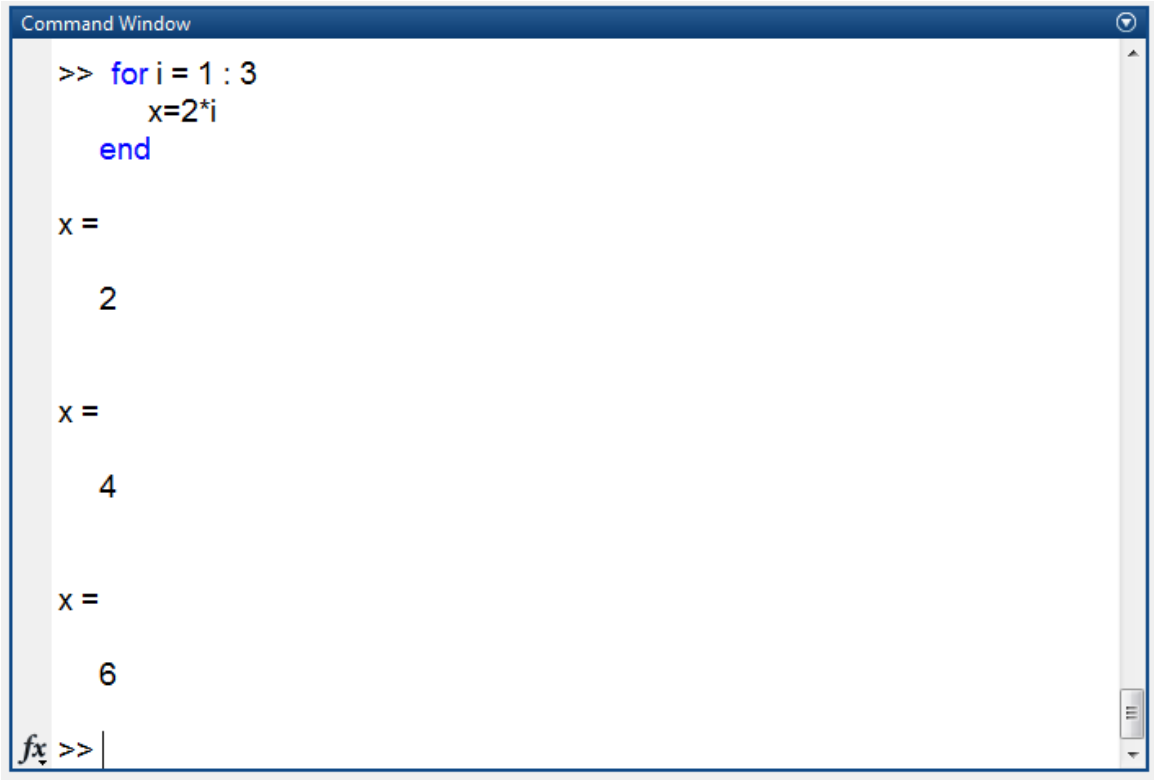
ولاحظ الخرج، (ملاحظة تعليمة rand تقوم بتوليد قيم عشوائية لذلك عند كل تنفيذ للتعليمة ستحصل على قيم مختلفة للمصفوفة الأولى).



- إنشاء عداد التكرار ضمن حلقة for.

مثال: اكتب ضمن Command Window ما يلي:

```
for i = 1 :3
x = 2*i
end
```

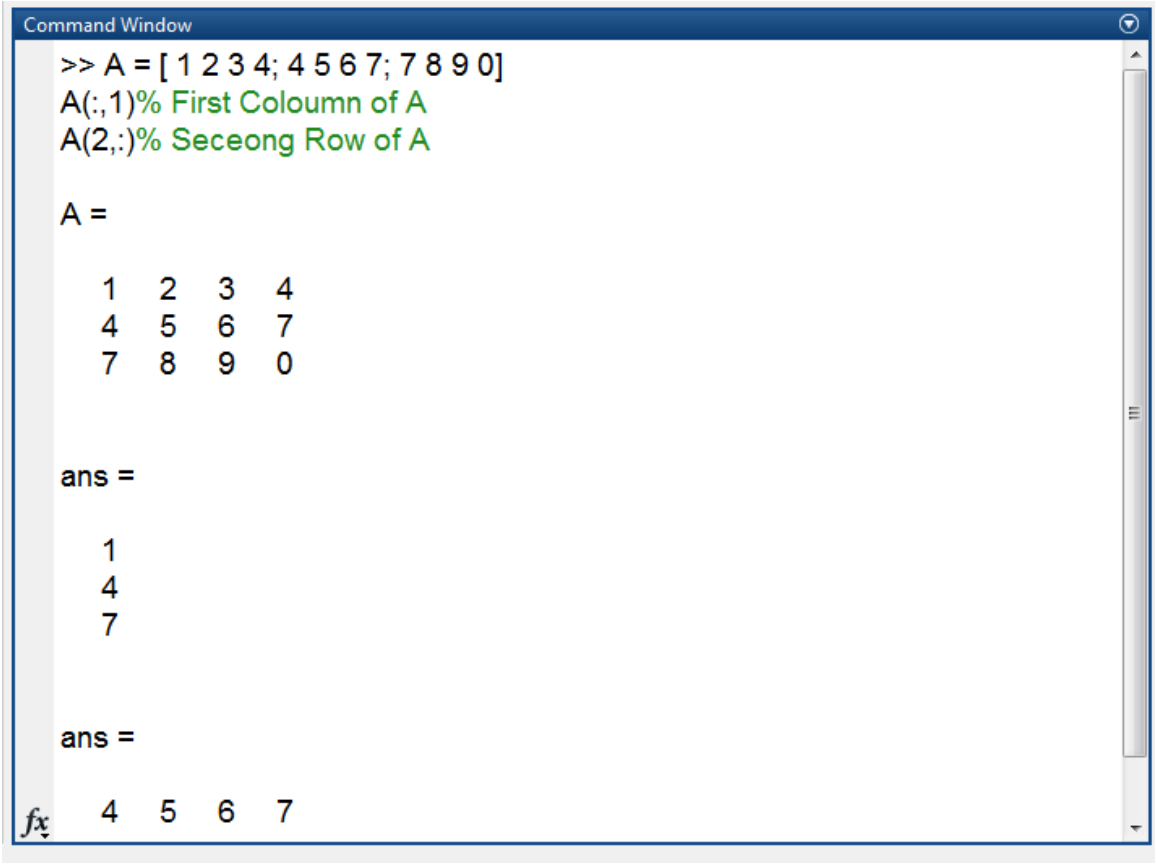


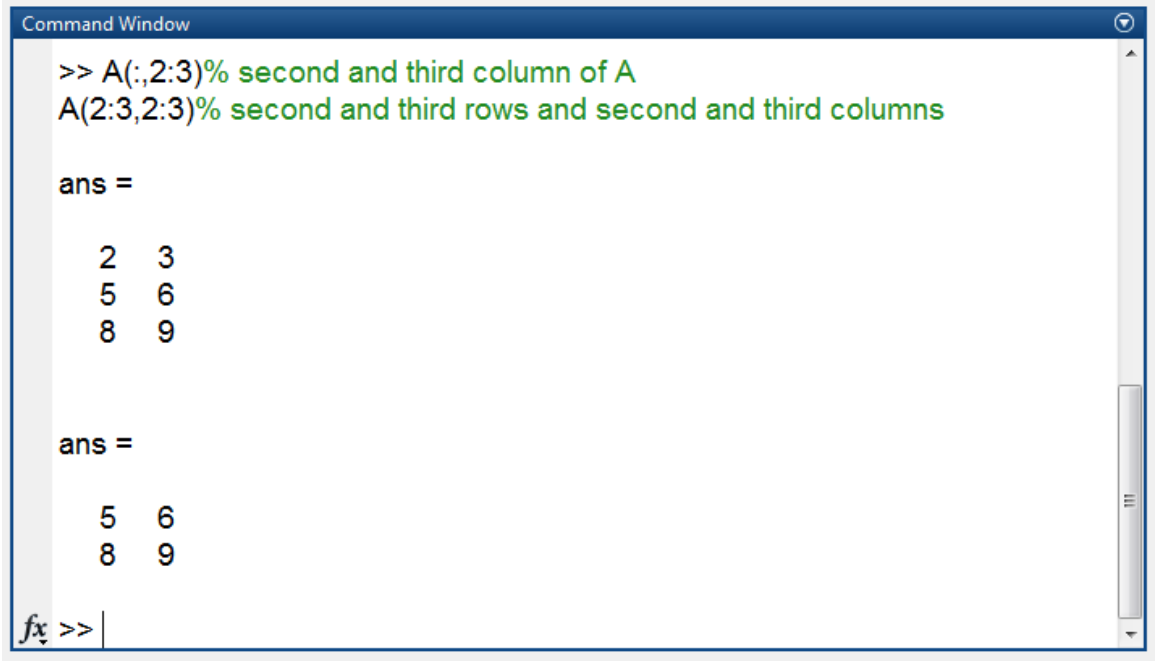
- تحديد مجال من القيم أو سطر أو عمود ضمن مصفوفة، صفيفة.

مثال: نعرف المصفوفة A على الشكل التالي، وننفذ ما يلي A(:,1) والتي تفيد في قراءة عناصر العمود الأول من المصفوفة A، و A(2,:) والتي تفيد في عناصر السطر الثاني من المصفوفة A، A(:,2:3) من أجل قراءة العمود الثاني والثالث من المصفوفة A، A(2:3,2:3) من أجل قراءة العمود الثاني والثالث و السطر الثاني والثالث من المصفوفة A.

- A = [1 2 3 4; 4 5 6 7; 7 8 9 0]

- A(:,1)% First Coloumn of A
- A(2,:)% second Row of A
- A(:,2:3)% second and third column of A
- A(2:3,2:3)% second and third rows and second and third columns





- تقوم بقراءة جميع عناصر المتحول وتحويله إلى شعاع عمود سواء كان المتحول هو سطر أو عمود أو مصفوفة.

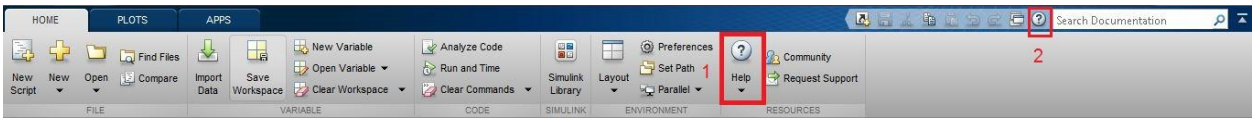
كما في المثال التالي:

عرّف A1=[1, 2, 3, 4; 5, 6, 7,8]; A2=[1 2 3 4]; ثم عرّف B1= A1(:) و B2=A2=(:)

```
Command Window
>> A1 = [ 1, 2, 3, 4; 5, 6, 7,8];
>> A2 = [ 1 2 3 4];
>> B1= A1(:)

B1 =

     1
     5
     2
     6
     3
     7
     4
     8

>> B2=A2(:)

B2 =

     1
     2
     3
     4
```

للاستعلام أكثر عن المؤثر colon استخدم MATLAB Help وفق تسلسل البحث التالي:

ادخل إلى HELP ضمن toolstrip جزء home قسم resources واختر Documentation هو رقم 1 بالصورة أو اضغط على الاختصار هو رقم 2 بالصورة



ادخل إلى MATLAB ثم Language Fundamentals ثم Matrices and Arrays ثم Indexing وادخل إلى Colon.

بعد النتهاء من الحديث عن المؤثر الهام colon سنعرض في هذه الفقرة المؤثرات operators التي تستخدم ضمن برمجية MATLAB إضافةً لوظيفتها، تقسم ضمن خمسة فئات وهي:

1. مؤثرات حسابية Arithmetic Operators
2. مؤثرات علاقاتية Relational Operators
3. مؤثرات منطقية Logical Operators
4. عمليات على مستوى البت Bit-Wise Operations

5. عمليات على المجموعات Set Operations

مؤثرات حسابية Arithmetic Operators

Operator	Description
+	Addition
-	Subtraction
.*	Multiplication
./	Right division
.\	Left division
+	Unary plus
-	Unary minus
:	Colon operator
.^	Power
.'	Transpose
'	Complex conjugate transpose
*	Matrix multiplication
/	Matrix right division
\	Matrix left division
^	Matrix power

تقسم المؤثرات الحسابية ضمن MATLAB إلى قسمين من العمليات هما:

- العمليات على الصفيفة Array Operations والتي تكون على مستوى العناصر أي عنصر لعنصر element-by-element.
- العمليات على مستوى المصفوفة Matrix Operations.

يمكن استخدام هذه المؤثرات الحسابية لإجراء الحسابات العدّدية، مثل جمع عددين أو شعاعين، ضرب مصفوفتين، رفع عناصر صفيفة إلى قوة معينة.

يجب الانتباه إلى الاختلاف بين النوعين السابقين حيث أن العمليات على المصفوفة تتبع قواعد الجبر الخطي، في حين أن العمليات على الصفيفة تنفذ على مستوى عنصر لعنصر ويمكن استخدامها على الصفيفات متعدّدة الأبعاد. يستخدم المحرف (.) للتمييز بين العمليات على الصفيفات والعمليات على المصفوفات.

ملاحظة: لا يوجد اختلاف بين عمليات الطرح والجمع بين الصفيفة والمصفوفة لذلك من غير الضروري استخدام المحرف . مع عملياة الجمع + أي لا نستخدم .+ بل نستخدم + فقط.

سنبدأ الشرح عن العمليات على الصفيفة يوضح الجدول التالي المؤثرات إضافةً إلى الهدف منها ووصف المؤثر إضافةً لأسماء التعليمات أو المفاهيم الموجودة ضمن MATLAB HELPمن أجل الاطلاع بشكل أوسع على عمل التعليمة وشرح عن المؤثر إضافةً للاستفادة من الأمثلة الموجودة ضمن HELP.

Operator	Purpose
+	Addition
+	Unary plus
-	Subtraction
-	Unary minus
.*	Element-wise multiplication
.^	Element-wise power
./	Right array division
.\	Left array division
.'	Array transpose

Description	Reference Page
A+B adds A and B.	plus
+A returns A.	uplus
A-B subtracts B from A	minus
-A negates the elements of A.	uminus
A.*B is the element-by-element product of A and B.	times
A.^B is the matrix with elements A(i,j) to the B(i,j) power.	power
A./B is the matrix with elements A(i,j)/B(i,j).	rdivide
A.\B is the matrix with elements B(i,j)/A(i,j).	ldivide
A.' is the array transpose of A. For complex matrices, this does not involve conjugation.	transpose

ملاحظة هامة: العمليات التالية صحيحة من أجل صفيفات بنفس البعد (equal dimensions).

1. + مؤثر الجمع، – مؤثر الطرح:

نعرف شعاعين ;[2 2 2 2] =B ;[1 1 1 1] =A أوجد ناتج A+B, B-A.

2. / مؤثر القسمة لكن يجب التفريق بين المؤثر / و \ لدينا المثال التالي:

اكتب 5/4 النتيجة هي 1.2500

أما 5\4 النتيجة هي 0.8000

أي المؤثر / يعني أن العدد إلى يسار المؤثر هو المقسوم والعدد إلى يمين المؤثر هو المقسوم عليه ويمكن تجريبها أيضاً على أشعة أو صفيفات.

3. *. مؤثر الضرب يقوم بضرب كل عنصر من الصفيفة بالعنصر المقابل له/ المثال التلي يوضح الفكرة التالية:

عرّف ;[1 2; 1 2] = B ;[1 2 ; 3 4] = A ونلاحظ أن الخرج هو كالتالي:

أما القسم الثاني من العمليات هو العمليات على المصفوفة يوضح الجدول التالي المؤثرات إضافةً إلى الهدف منها ووصف المؤثر إضافةً لأسماء التعليمات أو المفاهيم الموجودة ضمن MATLAB HELP من أجل الاطلاع بشكل أوسع على عمل التعليمة وشرح عن المؤثر إضافةً للاستفادة من الأمثلة الموجودة ضمن HELP.

Operator	Purpose
*	Matrix multiplication
/	Matrix right division
\	Matrix left division
^	Matrix power
'	Complex conjugate transpose

Description	Reference Page
C = A*B is the linear algebraic product of the matrices A and B. The number of columns of A must equal the number of rows of B.	mtimes
x = B/A is the solution to the equation xA = B. Matrices A and B must have the same number of columns. In terms of the left division operator, B/A = (A'\B')'.	mrdivide
x = A\B is the solution to the equation Ax = B. Matrices A and B must have the same number of rows.	mldivide
A^B is A to the power B, if B is a scalar. For other values of B, the calculation involves eigenvalues and eigenvectors.	mpower
A' is the linear algebraic transpose of A. For complex matrices, this is the complex conjugate transpose.	ctranspose

* مؤثر جداء المصفوفات والذي يتم وفقاً للعلاقة التالية $C_{i,j} = \sum_{k=1}^{N} A_{i,k}.B_{k,j}$

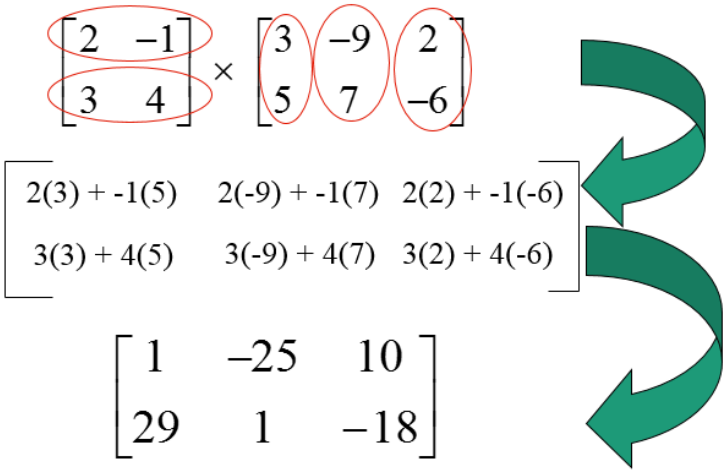
سنسعى في الشرح التالي التمييز بين المؤثر *. و * حيث وجدنا أن المؤثر *. يقوم بضرب كل عنصر من الصفيفة مع العنصر المقابل له بشرط أن تكون الصفيفات بنفس الأبعاد، اما بالنسبة لضرب المصفوفات فيجب أن يتحقق شرط الأبعاد الداخلية inner dimensions من أجل إجراء عملية الضرب.

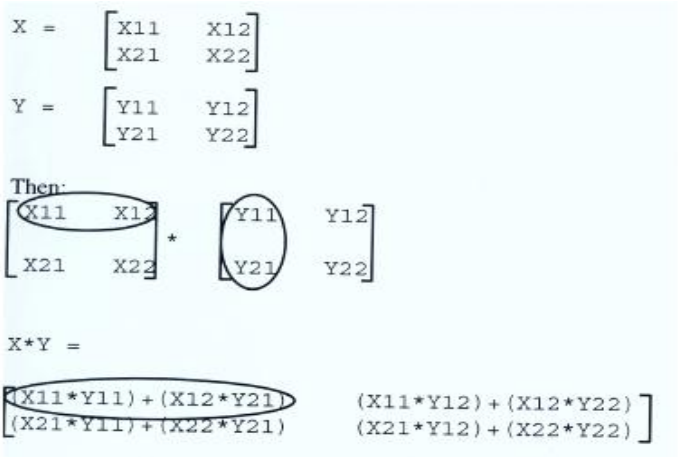
الخطوات من أجل ضرب مصفوفتين :

1. يجب أن يكون عدد أعمدة المصفوفة الأولى مساوي لعدد أسطر المصفوفة الثانية.
2. عندها يتم تطبيق العلاقة التالية $C_{i,j} = \sum_{k=1}^{N} A_{i,k}.B_{k,j}$ حيث يتم ضرب العناصر من كل سطر من المصفوفة الأولى بالعناصر من كل عمود من المصفوفة الثانية، ثم يتم جمع هذا ناتج ضرب كل عنصر مع العنصر المقابل له عند الانهاء من السطر مع العمود المقابل له.

المثال التالي للتوضيح: لدينا أولاً عدد أعمدة المصوفة الأولى يساوي 2 وهو مساوٍ لعدد أسطر المصفوفة الثانية ومنه يمكن تنفيذ عملية الضرب.

العنصر الأول في المصفوفة الناتجة هو حاصل الجداء السلمي للسطر الأول من المصفوفة الأولى مع العمود الأول من المصفوفة الثانية حيث يتم ضرب كل عنصر بمقابله وجمع النتيجة وهكذا من أجل باقي عناصر المصفوفة الناتجة والتي تكوت بأبعاد هي الأول هو عدد أسطر المصفوفة الأولى و الثاني هو عدد أعمدة المصفوفة الثانية.





مثال: لنقم بتوليد المصفوفتين التاليتين

A= [1 2; 3 4]; B= [3 0; 2 4]; ولنوجد المصفوفات التالية: C= A.*B و D= A*B.

يمكن استخدام help للاطلاع على عمل هذه المؤثرات بشكل أكبر. ابحث وفق المسار التالي

Matlab → language fundamaentals → operators and elementary operations → arithmetic.

مثال على استخدام المصفوفات في حل جملة المعادلات.

لتكن لدينا جملة المعدلات التالية، يمكن تمثيلها

$$\begin{aligned} 3x + 2y - z &= 1 \\ 2x - 2y + 4z &= -2 \\ -x + \tfrac{1}{2}y - z &= 0 \end{aligned}$$

يمكن تمثيلها بالشكل المصفوفاتي أي Ax=b حيث A هي مصفوفة، x هو شعاع المجاهيل و b هو شعاع الأمثال المعطاة.

نقوم بتشكيل المصفوفة A حيث عناصرها هي:

A= [3 2 −1; 2 −2 4; −1 0.5 −1]; b= [1; −2; 0]; ثم نقوم بإيجاد شعاع المجاهيل حيث يمكن إيجاد الشعاع x= [t; v; u] بثلاثة طرق متكافئة إلا أن فكرة الحل وحيدة وهي إيجاد مقلوب المصفوفة A وضربها بالشعاع b.

الطريقة الأولى باستخدام تعليمة inv والتي تفيد في إيجاد مقلوب مصفوفة.

الطريقة الثانية باستخدام المؤثر ^ أي power وهو الرفع إلى قوة نقوم برفعه إلى القوة 1− من أجل إيجاد المقلوب على اعتبار ان المؤثر يتعامل مع المصفوفة.

الطريقة الثالثة هي استخدام المؤثر \ حيث نقوم بتقسيم الشعاع b على المصفوفة A وهذا ممكن لأن هذا المؤثر معرّف من أجل العمليات على المصفوفات.

```
Command Window
>> A= [3 2 -1; 2 -2 4; -1 0.5 -1]; b= [1;-2;0];
x=inv(A)*b
y=(A^-1)*b
z=A\b

x =

   1.0000
  -2.0000
  -2.0000


y =

   1.0000
  -2.0000
  -2.0000


z =

   1.0000
  -2.0000
  -2.0000

fx >>
```

تمرين: قم بإيجاد حل جملة المعادلات التالية.

$$2x+3y+5z+t=1$$
$$4x+7y+9z=5$$
$$x+6y+2t=0$$
$$x+y+z-t=5$$

مؤثرات علاقاتية Relational Operators

تعمل هذه المؤثرات على مستوى عناصر الصفيفة بشكل عام شرط أن تكون بأبعاد متساوية، أي ان المقارنة تتم على مستوى عنصر لعنصر element-by-element. يكون خرج عملية المقارنة هو متحول منطقي logical أي هو عبارة عن 1 إذا كانت نتيجة المقارنة صحيحة (يدل على true) أو 0 إذا كانت نتيجة المقارنة خاطئة (يدل على false)

Operator	Description
<	Less than
<=	Less than or equal to
>	Greater than
>=	Greater than or equal to
==	Equal to
~=	Not equal to

يمكن إضافة حقل للجدول هو التفسير باللغة العربية لهذه المؤثرات

> أصغر من <= أصغر أو يساوي < أكبر من >= أكبر أو يساوي == للمساواة ~= عدم المساواة

مثال:

```
Command Window
>> A = 4*ones(3,3)

A =

   4   4   4
   4   4   4
   4   4   4

>> A >= [ 1 2 3; 4 5 6; 7 8 1]

ans =

   1   1   1
   1   0   0
   0   0   1

fx >>
```

مؤثرات منطقية Logical Operators

يتيح MATLAB للمستخدم نوعين من المؤثرات المنطقية:

- Element-wise (مستوى العناصر) تعمل على عناصر الصفيفات المنطقية logical arrays.

لناخذ المثال التالي من أجل معرفة عمل هذه المؤثرات وما هي التعليمات أو المحارف الخاصة بها ضمن اللغة.

نعرف الشعاعين التاليين ;B= [1 1 0 0 1] ;A = [0 1 1 0 1] يبين الجدول التالي المؤثرات المنطقية على مستوى العنصر ونتيجة تطبيق هذه المؤثرات على الشعاعين السابقين.

Operator	Description	Example
`&`	Returns 1 for every element location that is `true` (nonzero) in both arrays, and 0 for all other elements.	`A & B = 01001`
`|`	Returns 1 for every element location that is `true` (nonzero) in either one or the other, or both arrays, and 0 for all other elements.	`A | B = 11101`
`~`	Complements each element of the input array, `A`.	`~A = 10010`
`xor`	Returns 1 for every element location that is `true` (nonzero) in only one array, and 0 for all other elements.	`xor(A,B) = 10100`

يمكن التعرف أكثر على التوابع المتوفرة ضمن MATLAB وعلى المؤثرات عن طريق Help. قم بالبحث عن logical operators و logical operators: element-wise و logical-operators: short-circuit و يمكن أيضاً البحث عن operators يوجد ضمنها فقرة خاصة ب logical operators.

- Short-circuit تعمل على التعابير المنطقية التي تحتوي على قيم السلمية scalar and logical expressions.

هذه المؤثرات هي

Operator	Description
&&	Returns logical 1 (true) if both inputs evaluate to true, and logical 0 (false) if they do not.
||	Returns logical 1 (true) if either input, or both, evaluate to true, and logical 0 (false) if they do not.

يمكن استخدام help للاطلاع على عمل هذه المؤثرات بشكل أكبر.

عمليات على مستوى البت Bit-Wise Operations

تفيد في إجراء عمليات منطقية على مستوى البت مثل and, or, xor وإيجاد المكمل complement والقيام بعملية الأزاحة shift و ....، للاطلاع على المزيد عن هذا الموضوع استخدم help.

نعرض المثال التالي:

يتم تعريف قيمتين A,B ومن ثم يتم استخدام بعض العمليات المنطقية على مستوى البت، الخرج موضّح أيضاً.

```
A = 28;                 % binary 00011100
B = 21;                 % binary 00010101
```

Function	Description	Example
bitand	Returns the bit-wise AND of two integer arguments.	bitand(A,B) = 20 (binary 00010100)
bitor	Returns the bit-wise OR of two integer arguments.	bitor(A,B) = 29 (binary 00011101)
bitcmp	Returns the bit-wise complement, assuming the second argument specifies the integer data type of the first argument. By default, MATLAB treats double values as uint64 integers.	bitcmp(A,'int8') = -29 (binary 11100011)
bitxor	Returns the bit-wise exclusive OR of two integer arguments.	bitxor(A,B) = 9 (binary 00001001)

مثال: قم بتعريف مايلي:

a = 60; % 60 = 0011 1100

b = 13; % 13 = 0000 1101

c = bitand(a, b) % 12 = 0000 1100

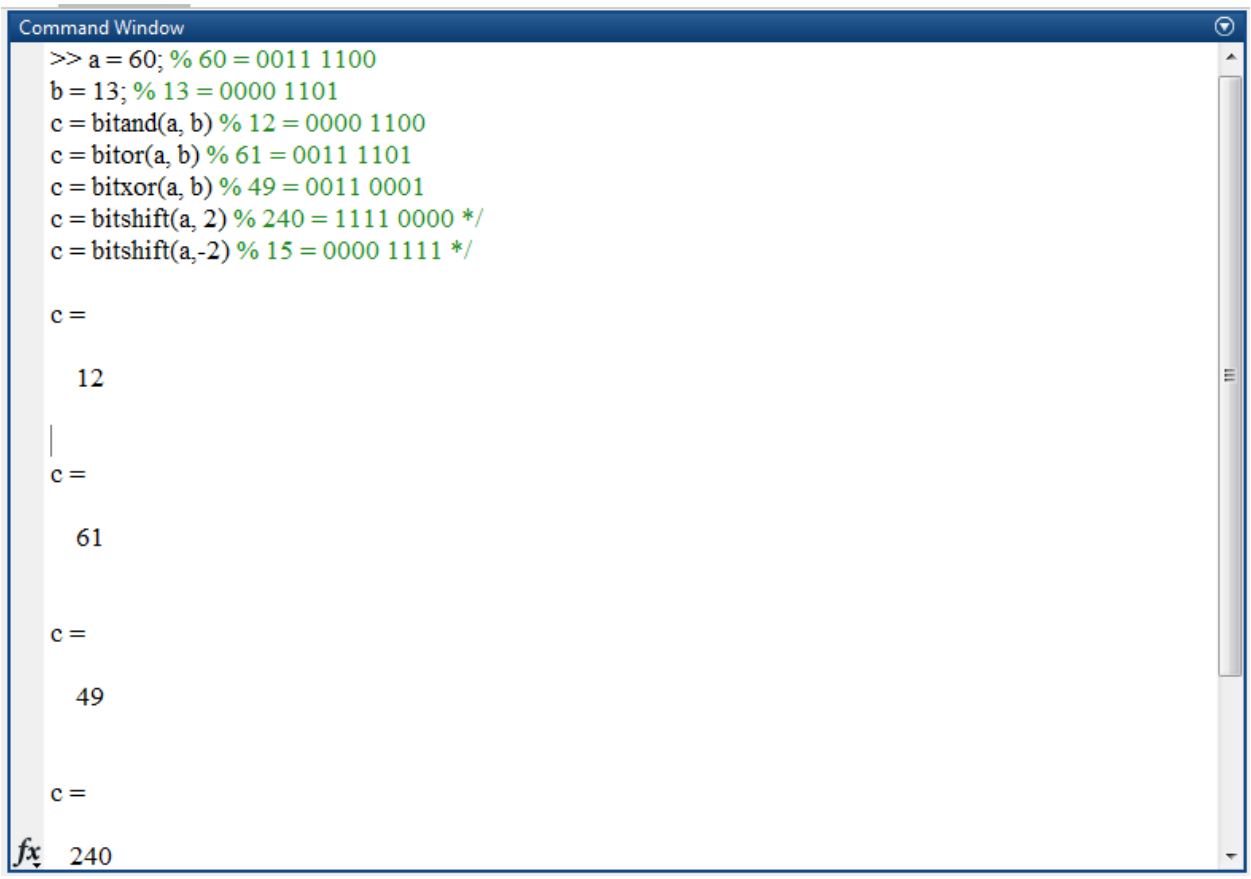
c = bitor(a, b) % 61 = 0011 1101

c = bitxor(a, b) % 49 = 0011 0001

c = bitshift(a, 2) % 240 = 1111 0000/*

c = bitshift(a,-2) % 15 = 0000 1111/*

الخرج على الشكل التالي:



c =

15

ملاحظة: يرجى دمج هذه الصورة مع الصورة السابقة لأنها نتيجة الخرج.

عمليات على المجموعات Set Operations

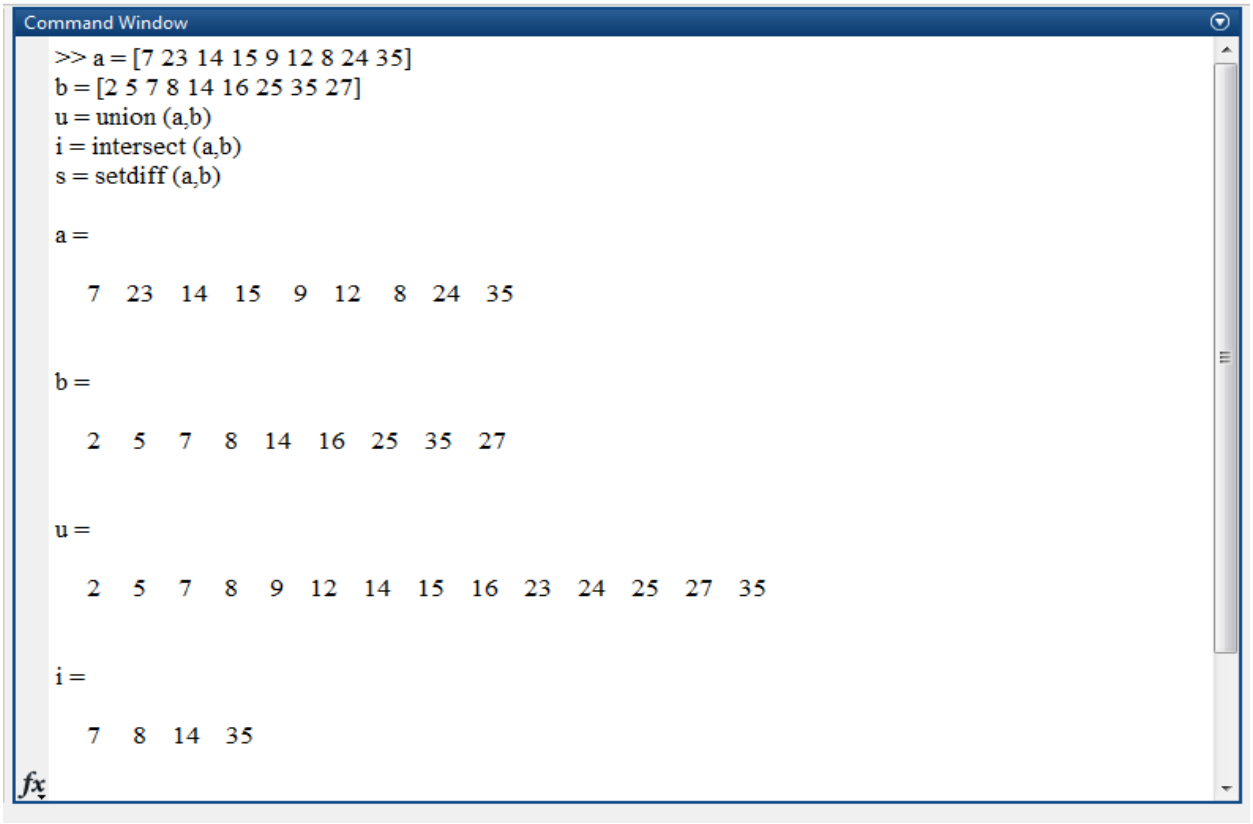
وهي تشمل عمليات التقاطع، الاجتماع، الانتماء إلى مجموعة، الفرق إضافةً إلى العديد من التوابع الممكن استخدامها في هذا المجال، يبين الشكل التالي صورة لها للاطلاع أكثر على عملها استخدم help وابحث عن set operations وقم بقراءة عمل كل تابع عند الحاجة للعمل مع مثل هذه التوابع.

MATLAB Functions

intersect	Find set intersection of two arrays
ismember	Array elements that are members of set
issorted	Determine whether set elements are in sorted order
setdiff	Find set difference of two arrays
setxor	Find set exclusive OR of two arrays
union	Find set union of two arrays
unique	Find unique values in array

مثال:

```
a = [7 23 14 15 9 12 8 24 35]
b = [2 5 7 8 14 16 25 35 27]
u = union (a,b)
i = intersect (a,b)
s = setdiff (a,b)
```




```
s =

   9   12   15   23   24
```

ملاحظة يرجى دمج هذه الصورة مع الصورة السابقة لأنها خرج للتنفيذ.

فهرسة المصفوفات Array indexing:

من أجل الوصول إلى عنصر معين أو مجموعة من العناصر ضمن شعاع، مصفوفة أو صفيفة لابد لنا من استخدام عملية الفهرسة indexing، هنا تجدر اذكر الملاحظة الأكثر أهمية وهو ان MATLAB لا يقبل أي دليل سابي أي أن الوصول للعنصر الأول ضمن شعاع أو مصفوفة ما هو عبر الدليل 1 وليس الدليل 0.

مثال: عرّف الشعاع التالي ;[3 2 1 0] =Ind من أجل الوصول للعنصر الأول وهنا قيمته 0 نكتب Ind(1) وإلا ستظهر رسالة خطأ كما في الشكل التالي:

```
Command Window
>> Ind= [ 0 1 2 3];
>> Ind(1)

ans =

    0

>> Ind(0)
Subscript indices must either be real positive integers or logicals.

fx >>
```

الوصول إلى عنصر

يتم الوصول إلى عنصر ضمن مصفوفة عن طريق تحديد ضمن اي سطر وعمود يقع العنصر أي على الشكل:

Matrix (row, coloumn)

لنقم بتوليد مصفوفة ببعد 5 × 5 باستخدام التعليمة magic. نكتب التعليمة التالية A= magic(5) قم بالاطلاع ضمن HELP على عمل التعليمة magic، ثم قم بإيجاد العنصر الواقع في السطر الرابع والعمود الثالث.

```
Command Window
>> A = magic (5)

A =

   17   24    1    8   15
   23    5    7   14   16
    4    6   13   20   22
   10   12   19   21    3
   11   18   25    2    9

>> x = A (4,3)

x =

   19

fx >>
```

هنالك طريقة أخرى للوصول إلى عنصر حيث ان MATLAB يقوم بتخزين المصفوفة ضمن الذاكرة بشكل تسلسلي أي أن أي مصفوفة تخزن على شكل عمود عدد أسطره هو عدد عناصر المصفوفة وعدد أعمدته 1. عناصر العمود الأول في المصفوفة الأصلية هي الأسطر الأولى ضمن المصفوفة ثم يأتي بعدها عناصر العمود الثاني وهكذا لنوضح ذلك بمثال.

مثال: لنقم بتوليد المصفوفة التالية ;[1, 2, 3; 8, 3, 6; 1, 2, 5] = A يتم تخرين المصفوفة التالية في الذاكرة على الشكل التالي

1

8

1

2

3

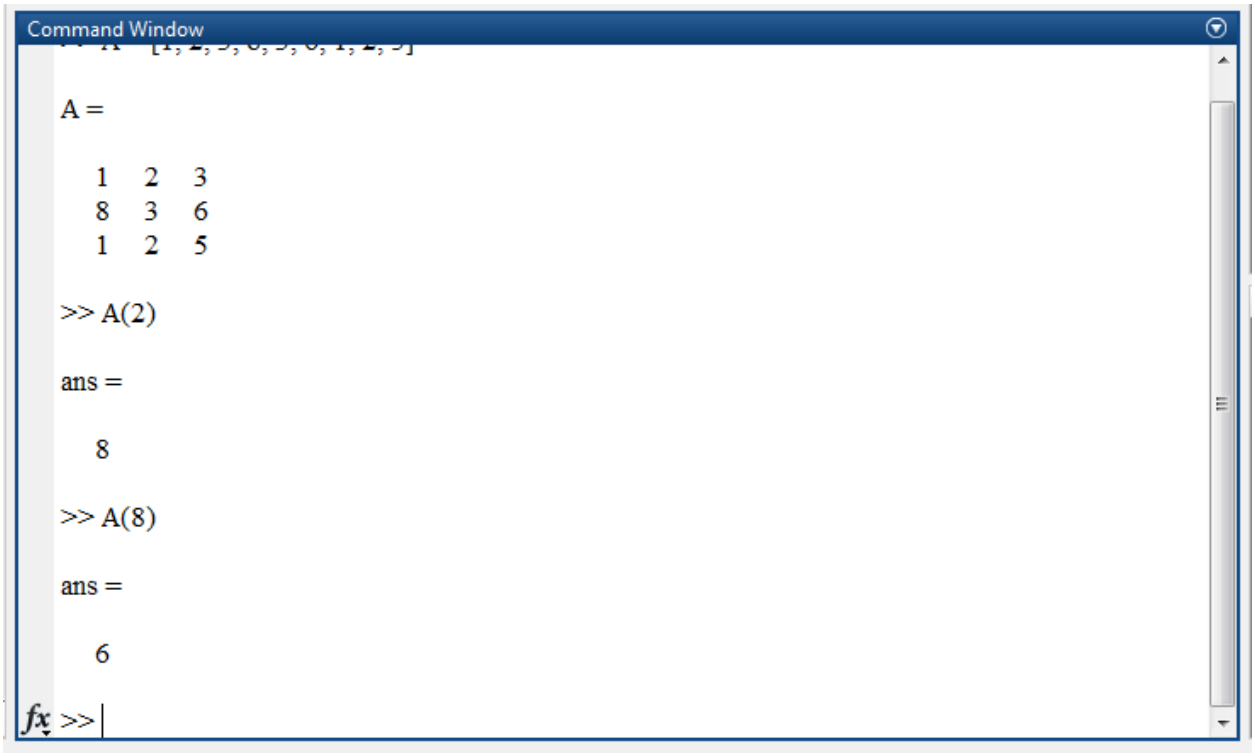
2

3

6

5

يمكن الوصول إلى العنصر 8 والذي يقع ضمن السطر الثاني العمود الأول عن طريق كتابة A(2)، أما العنصر 6 الواقع في السطر الثاني العمود الثالث فترتيبه هو العنصر الثامن أي نكتب A(8)




```
MAT =

   1   2   3
   4   5   6
   9   8   7
```

1
4
9
2
5
8
3
6
7

https://www.youtube.com/watch?v=tqjZ80PwqBU
https://www.youtube.com/watch?v=q8lUOI0uqTo

الأسئلة

1. ما هي قيمة المتحول ans بعد تنفيذ الرماز التالي
   a. 1+6
   b. X =1+6
   c. E=X+2
   d. ؟؟ 4+5

مساعدة: قراءة فقرة عمليات حسابية بسيطة و فقرة محارف وقيم خاصة

2. كلمة NAN هي اختصار لأي كلمة وما دلالتها ومتى يكون خرج العملية الحسابية هو القيمة NAN ؟؟

مساعدة: قراءة فقرة محارف وقيم خاصة

3. عند تعريف المتحول x=1; ما هو نمط المتحول؟؟

مساعدة: قراءة فقرة مقدمة عن المتحولات

4. ما هو الفرق بين عمل التعليمة clc والتعليمة clear all

مساعدة: قراءة فقرة مقدمة عن المتحولات

5. قم بتعريف متحول سلمي وسمّه x وأسند له القيمة 10 وأضف تعليق إلى جانبه وهو this variable is the answer of question 4 وضمن التعريف لا يجب عند تنفيذ التعليمة أن تظهر النتيجة في command window.

مساعدة: قراءة فقرة كيف نعرف متحول سلمي

6. قم بتعريف شعاع باسم vect ببعد 30 × 1 يحتوي على كافة الأعداد الفردية ضمن المجال [1-60] بحيث عند تنفيذ التعليمة يتم عرض القيم في command window كيف يمكن معرفة طول الشعاع اكتب التعليمة اللازمة لذلك

مساعدة: قراءة فقرة كيف نعرف شعاع

7. قم بتعريف المصفوفات التالية:
   - مصفوفة مربعة ببعد 2 × 2
   - مصفوفة مربعة ببعد 5 × 4 باستخدام التعليمة zeros
   - مصفوفة مربعة ببعد 2 × 3 باستخدام التعليمة ones

- مصفوفة مربعة ببعد 2×5 باستخدام التعليمة rand

مساعدة: قراءة فقرة كيف نعرف مصفوفة

8. كيف ممكن معرفة بعد هذه المصفوفة وحاصل جمع عناصر هذه المصفوفة اكتب الرماز البرمجي اللازم، المصفوفة هي

A= [2 5 6; 1 1 3];

مساعدة: قراءة فقرة كيف نعرف مصفوفة

9. في حال قمنا بتعريف المصفوفات التالية

a=[1,3; 2 ,4;3,4] b=[1,1;2,2] c=[0,1] d=[4;5;6]

كيف يتم تعريف صفيفة تحتوي كافة المصفوات السابقة ؟؟ و كيف من الممكن معاينة العنصر c

مساعدة: قراءة فقرة كيف نعرف صفيفة

10. اكتب الرماز البرمجي الالذي يقوم بتنفيذ ما يلي: في حال قمنا بتعريف المصفوفة التالية

A = [0 3 3 4; 4 2 1 7; 7 4 9 0; 47 86 19 10]

كيف يمكن قراءة العمود الثالث كاملاً من المصفوفة

كيف يمكن قراءة السطر الثالث كاملاً من المصفوفة

كيف يمكن قراءة العنصر الأول والثاني ضمن السطر الرابع من المصفوفة

مساعدة: قراءة فقرة المؤثرات ضمن MATLAB وتحديداً المؤثر colon.

11. في حال لدينا المصفوفتين التاليتين A,B المعرفتين وفق ما يلي

A= [1 , 1; 2, 2] B= [2,3;4,0]

ما قيمة اخرج العليمات التالية

X= A+3

Y= A^2

Z=2*B

C=A*B

D=A.*B

مساعدة: قراءة فقرة المؤثرات ضمن MATLAB وتحديداً المؤثرات الحسابية

12. قم بإيجاد حل جملة المعادلات التالية.

$$2x + 3y + 5z + t = 1$$
$$4x + 7y + 9z = 5$$
$$x + 6y + 2t = 0$$
$$x + y + z - t = 5$$

مساعدة: قراءة فقرة المؤثرات ضمن MATLAB وتحديداً المؤثرات الحسابية وبشكل أدق التمرين الذي طلب تحضيره

13. ما خرج التعليمات التالية، في حال كان أحدها يعطي خطأ يرجى الإجابة بكلمة error فقط.

X = [1 ,2 ; 3 ,4; 5 ,6]

X(0)

X(6)

X(5)

X(3,2)

X(4,7)

الإجابات

1. 9

2. NAN وهي اختصار ل Not-a-Number للدلالة على أنها ليست قيمة عددّية، وهي عبارة عن النتيجة من عمليات غير معرفة نتيجتها نذكر منها:

- أي عملية حسابية على NAN
- خرج عملية $\infty - \infty$ أو $+\infty$ أو $0 * \infty$ أو $\frac{0}{0}$ أو $\frac{\infty}{\infty}$

3. النمط double 64-bit

4. الأولى clc تقوم بمسح التعليمات الموجودة ضمن command window دون مسح المتحولات أي تبقى ضمن workspace الثانية clear all تقوم بمسح جميع المتحولات التي تم تعريفها

5. x=10; % this variable is the answer of question 4

6. vect = 1:2: 60 ويتم معرفة الطول عن طريق كتابة length(vect)

7. يترك تصحيح الإجابة للأستاذ المشرف على المقرر.

8. size(A), sum(A)

9. arr={a,b;c,d} أما من أجل معاينة العنصر c نكتب arr{2,1}

10. من أجل قراءة العمود الثالث نكتب A(:,3)

من أجل قراءة السطر الثالث نكتب A(3,:)

من أجل قراءة العنصر الأول والثاني ضمن السطر الرابع A(4,1:2)

11.
```
X = [4 4; 5 5];
Y= [3 3; 6 6];
Z= [4 6; 8 0];
C= [6 3; 12 6];
D= [2 3; 8 0];
```

12.
```
a = [2 3 5 1
     4 7 9 0
     1 6 0 2
     1 1 1 -1],
b = [1 5 0 5],
x = a\b'
```

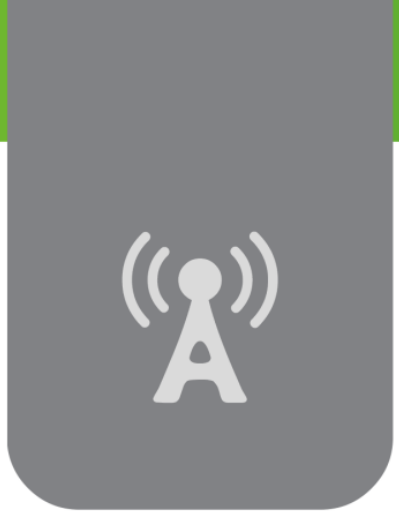
13. X(0) تعطي error

X(6) تعطي 6

X(5) تعطي 4

X(3,2) تعطي 6

X(4,7) تعطي error



الفصل الثالث: البرمجة في MATLAB® -1- (استخدام ملفات scripts والتوابع)

Programming in MATLAB® -1- (Use of Scripts files and Functions)

عنوان الموضوع:

البرمجة في MATLAB® -1- (استخدام ملفات scripts والتوابع)

Programming in MATLAB® -1- (Use of Scripts files and Functions)

الكلمات المفتاحية:

MATLAB، لواحق الملفات file extension، m file.، script، تابع function، محرر النصوص Text Editor، التوابع مجهولة الاسم Anonymous function، التوابع المحلية Local function، التوابع المتداخلة Nested function، التوابع الخاصة Private function.

ملخص:

يهتم هذا الفصل بتعلم البرمجة باستخدام لغة MATLAB، والتركيز على استخدام ملفات scripts والتوابع functions، حيث نبدأ بعرض بسيط للواحق الملفات ضمن MATLAB، لنبدأ بعدها بالتعرف على ملفات script، وملفات التوابع function والفروقات بين الملفات السابقة، ثم يتم ضمن هذا الفصل التعامل مع ملفات script من حيث الإنشاء، البرمجة، التنفيذ وكذلك الأمر بالنسبة للتوابع ثم ننهي الفصل بعرض أنواع التوابع ضمن لغة البرمجة MATLAB. ويتم تقديم أمثلة عديدة ضمن الفصل بهدف التعامل أكثر مع اللغة وتوضيح كافة الأفكار السابقة.

أهداف تعليمية:

يتعرف الطالب في هذا الفصل على:

- التعامل مع scripts و functions ضمن MATLAB.
- إنشا، حفظ وتنفيذ ملف script أو تابع function ضمن MATLAB.
- التعامل مع أنواع التوابع ضمن لغة MATLAB ومعرفة أهمية كل منها.

لواحق الملفات ضمن MATLAB

نعرض في هذه الفقرة لواحق الملفات التي يمكن تخرينها باستخدام برمجية MATLAB، وما يقابلها ضمن البرمجية، حيث يمكن أن تكون هذه الملفات على شكل توابع، scripts، صفوف، إنشاء Simulink model أو الحصول على أشكال أو .... .

ملاحظة: يوجد إضافةً للواحق التالية عدد من اللواحق لكن ليست ضمن اهتمامنا في هذا المقرر

اللاحقة	الاستخدام
.fig	MATLAB figure
.m	MATLAB code (function, script, or class)
.mat	MATLAB data (binary file for storing variables)
.prj	project file used by various solutions (packaged app/toolbox projects, MATLAB Compiler/Coder projects, Simulink projects)
.mdl	Simulink Model
.mdlp	Simulink Protected Model
.slx	Simulink Model
.slxp	Simulink Protected Model

Scripts Vs Functions

حتى الآن، وجدنا أنه يمكن استخدام بيئة MATLAB عوضاً عن الآلة الحاسبة. إلا أنّه في الواقع فإن إمكانيات هذه البرمجية هائلة، وخاصةً في مجال الحسابات العدّدية.

تم التعلم في الفصل السابق والفقرات السابقة كيفية إدخال التعليمات commands عن طريق Command Window، وهي ما تسمى بـ Line-command أو Line-statement لأنها تدخل على شكل تعليمة سطر وحيد. يتيح MATLAB أيضاً للمستخدم كتابة مجموعة من التعليمات ضمن ملف، ومن ثم تنفيذ الملف كاملاً، بشكل مشابه لكتابة التوابع Functions ضمن أي لغة برمجة.

وجدنا في فقرة لواحق الملفات ضمن MATLAB وجود لاحقة ".m" أو ما تسمى بـ M-files وهي مخصّصة لنوعين من ملفات البرامج ضمن هذه البيئة، وهما:

- Script
- Function

Scripts: ملفات script عبارة عن ملفات برامج بلاحقة m.، تعتبر أبسط أنواع البرامج ضمن MATLAB و يمكن ضمن هذه الملفات كتابة مجموعة من التعليمات المتعاقبة، والتي يرغب المبرمج بتنفيذها سويةً. لا تقبل ملفات Scripts وجود دخل input لها، ولا تقوم بإعادة أي خرج output، وعلى الرغم من ذلك فإن أي متحول يجري تعريفه خلال التنفيذ يبقى ضمن workspace ويمكن الاستفادة منه لاحقاً في حسابات أخرى.كما أنها تتعامل مع المعطيات الموجودة مسبقاً ضمن Workspace ويمكن من خلالها إنشاء وتعريف معطيات جديدة للتعامل معها. إضافةً لما سبق يمكن الحصول على رسومات بيانية باستخدام التعليمات المخصصة لذلك مثل plot.

لكل ملف Script اسم مضافاً له اللاحقة m.، مثلاً myclac.m، يمكن الاستفادة كثيراً من ملفات Script في إجراء عدد من الحسابات يتم استخدماتها بشكل متكرر من المستخدم ، فيقوم المستخدم بكتابتها ضمن Script وحفظها بهدف إعادة الاستعمال عند الحاجة.

التوابع Functions: ملفات التوابع عبارة عن ملفات برامج بلاحقة m. أيضاً، يسمح التابع بوجود دخل وإعادة خرج، المتحولات المعرّفة ضمن التابع عبارة عن متحولات محليّة local أي لا يمكن استخدامها إلّا ضمن التابع نفسه، ملفات التوابع عبارة عن مجموعة من التعليمات والحلقات والحالات الشرطية المتربطة مع بعضها بهدف تحقيق غرض معيّن، ضمن MATLAB يتم تعريف التوابع ضمن ملفات منفصلة، **ولكن يجب أن يكون اسم التابع مطابقاً لاسم الملف التي يتم حفظه.**

ملاحظة: يمكن للتوابع أن تقبل أكثر من دخل، وأن تعيد أكثر من خرج.

مقارنة بين scripts و functions:

ملفات البرامج ضمن MATLAB هي إما scripts تقوم ببساطة بتنفيذ مجموعة من التعليمات لا تقبل دخل ولا تعطي خرج أو functions تقبل بأن يكون لها متحولات دخل، وتعطي كنتيجةً خرج. كل من scripts و functions يحتويان على رماز برمجي مكتوب بلغة MATLAB، كما أنها كل منهما يجري تخزينه ضمن ملف بلاحقة m.، إن تنفيذ scripts سيتم بنفس الطريقة دوماً أما تنفيذ التوابع وخرجها فيعتمد على متحولات الدخل التي يقوم المستخدم بإدخالها مما يجعل التوابع أكثر عموميةً وفائدةً للمستخدم.

كيف يتم إنشاء new script ؟؟

في هذه الفقرة سنتعرف أكثر على ملفات scripts، رأينا أن ملفات Scripts تحتوي على عدد من السطور المتعاقبة التي تحتوي على تعليمات MATLAB أو توابع منجزة في مكاتب MATLAB أو حتى توابع قام المبرمج بتنجيزها مسبقاً.

من أجل إنشاء ملفات m. يمكن استخدام محرر النصوص text editor الخاص ببرمجية MATLAB أو أي محرر نصوص آخر مثل Notepad++، سنعرض فيما يلي الطرق الممكنة لإنشاء new script:

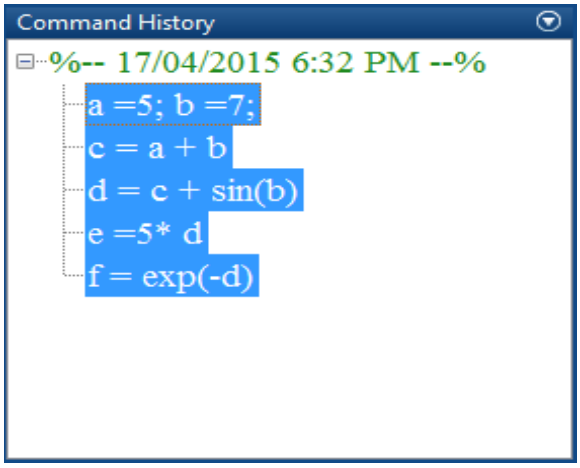
- من خلال Command History.
- من خلال شريط Toolstrip ثم نافذة Home.
- عن طريق استخدام التابع edit ضمن Command Window.

1. من خلال Command History: بعد كتابة مجموعة من التعليمات ضمن Command window وتنفيذها فإن نسخة من هذه التعليمات يتم حفظها في Command History يمكن تحديدها في النافذة السابقة والضغط بالزر اليميني واختيار "Create Script".

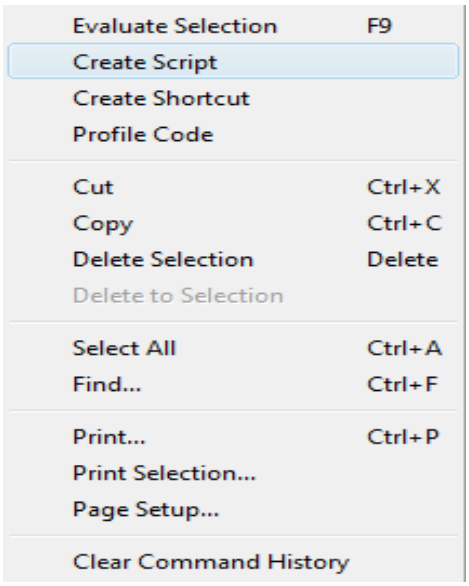
مثلاً اكتب ضمن Command Window التعليمات التالية

```
a =5; b =7;
c = a + b
d = c + sin(b)
e =5* d
f = exp(-d)
```

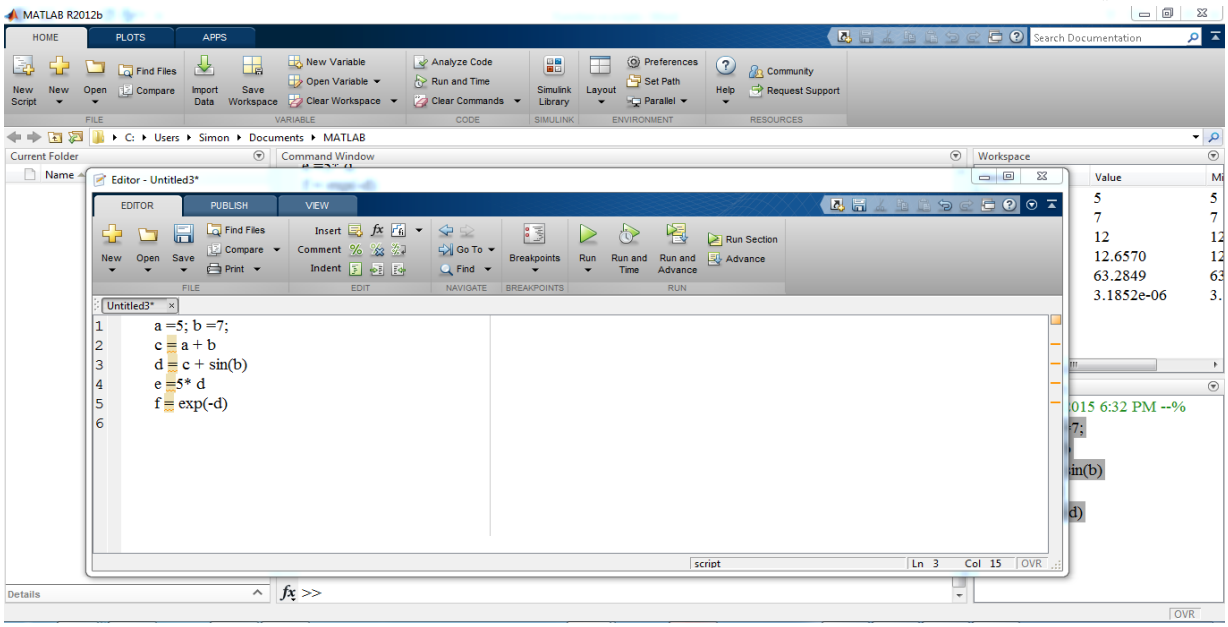
اذهب إلى Command History ستجد نسخة من هذه التعليمات، قم بتحديد التعليمات كما في الشكل



اضغط الزر اليميني ستجد عدة خيارات كما في الشكل التالي، قم باختيار Create Script.

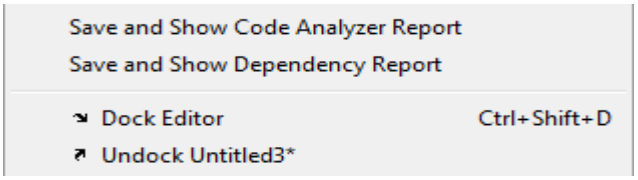


عند إنشاء Script يتم فتح واجهة جديدة تدعى Editor، وهذه الواجهة منفصلة عن الواجهات السابقة كما في الشكل التالي:

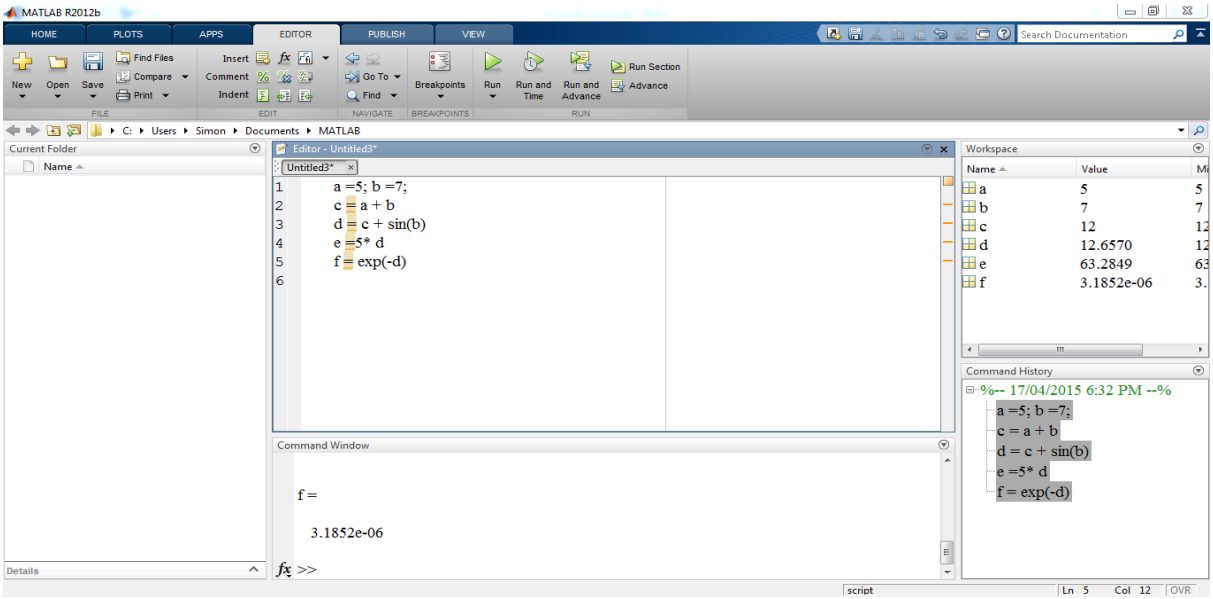


يمكن إعادة هذه الواجهة إلى الواجهة الرئيسية عن طريق استخدام الزر الموضح في الشكل التالي:

يعطي هذا الزر Show Editor Action عند الضغط عليه يظهر عدة خيارات قم باختيار Dock Editor



عند اختيار الخيار السابق ستحصل على الشكل التالي.



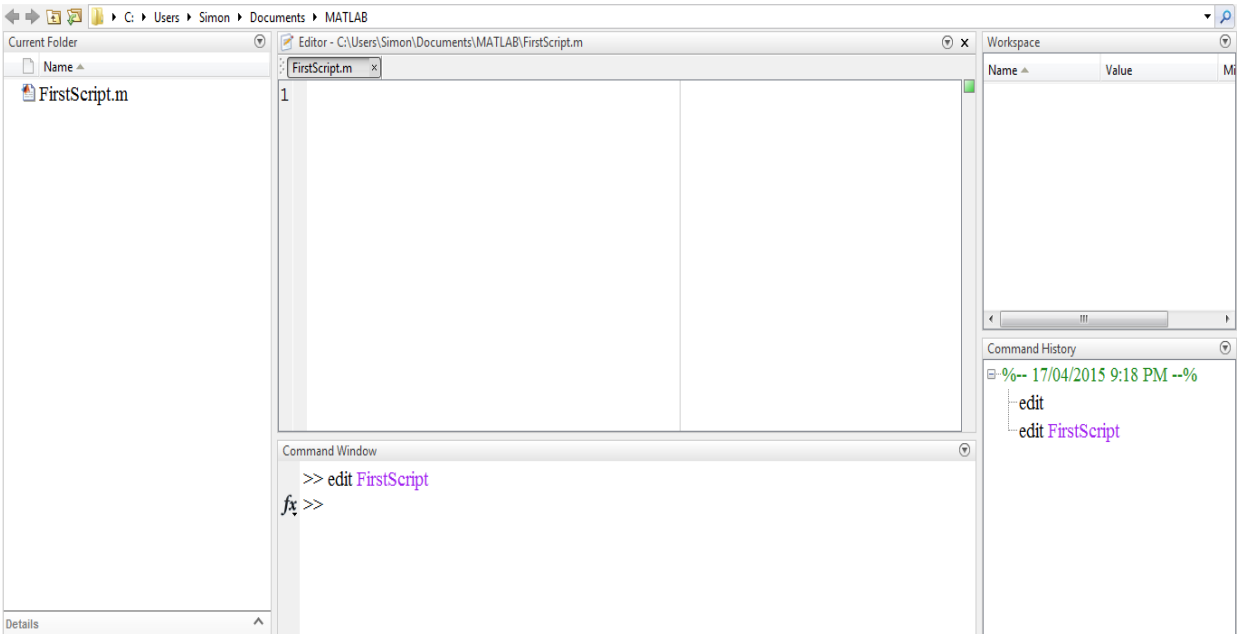
2. من خلال شريط Toolstrip ثم نافذة Home: ضمن Home ستجد زر New Script قم بالضغط على الزر السابق، أو قم بالضغط على الزر new ثم اختر script أو قم باستخدام Ctrl+N.

3. عن طريق استخدام التابع edit ضمن Command Window.

اكتب ضمن Command Window التعليمة التالية

```
edit FirstScript
```

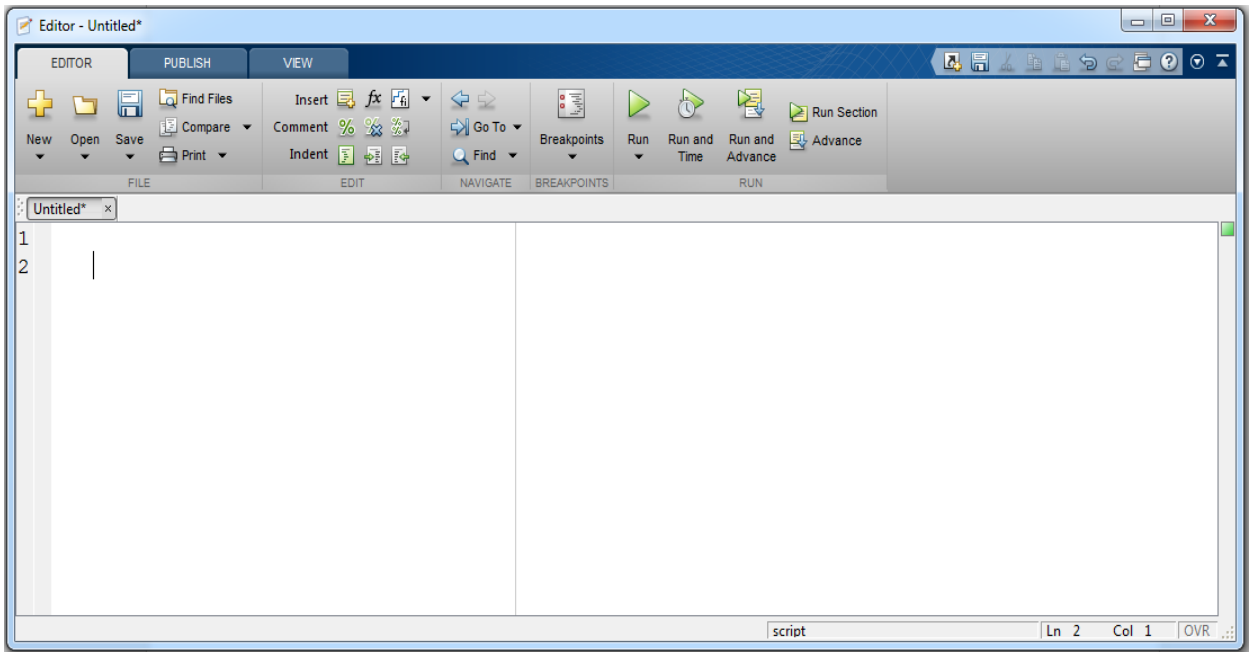
إن التعليمة السابقة ستقوم بإنشاء new script، وذلك حسب الاسم الذي تم كتابته كما في الشكل التالي:



ملاحظة: في حالة عدم تحديد اسم لملف Script فإن MATLAB يولد ملفاً جديداً باسم Untitled.

التعرف على محرر النصوص Editor ضمن MATLAB

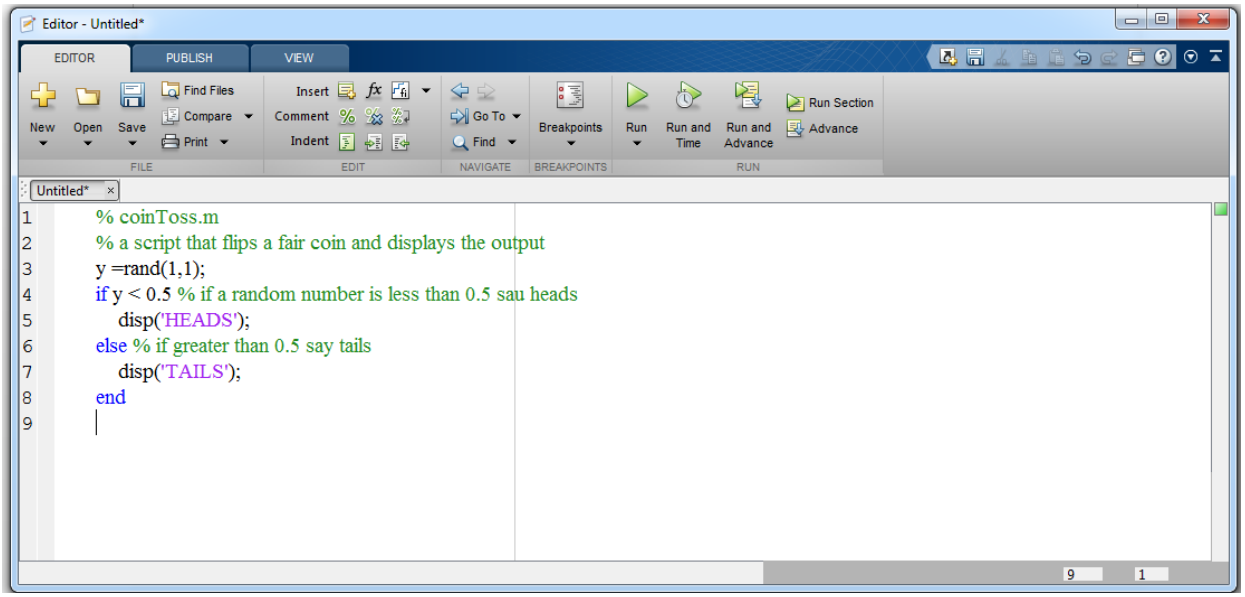
رأينا عند إنشاء new script يتم فتح النافذة التالية والتي تدعى Editor:



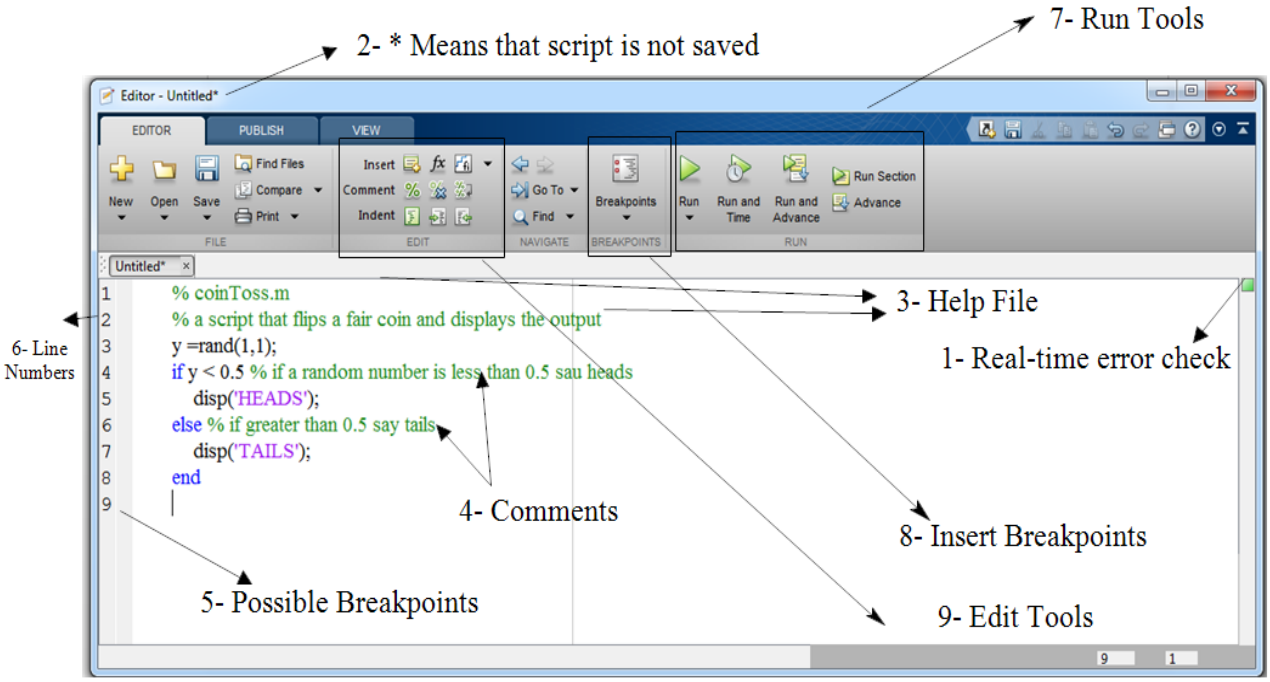
لنتعرف قليلاً على Editor والذي يتم ضمنه كتابة الرمازات البرمجية في لغة MATLAB. يبين الشكل التالي صورةً لل editor وقد تم كتابة رماز برمجي بسيط يقوم بإظهار كلمة Head أو Tail حسب قيمة المتحول العشوائي.

الرماز البرمجي هو:

```
%coinToss.m
%a script that flips a fair coin and displays the output
y =rand(1,1);
if y < 0.5 % if a random number is less than 0.5 sau heads
    disp('HEADS');
else % if greater than 0.5 say tails
    disp('TAILS');
end
```



يبين الشكل التالي الأجزاء الموجودة ضمن محرر النصوص:



إن الشكل السابق كما هو واضح يحتوي على عدة أجزاء سنقوم بشرحها:

1. إن العلامة السابقة تدل على عدم وجود خطأ error وذلك عند تلونها باللون الأخضر، اما في حالة وجود خطأ error أو تحذير warning يتم تغيير اللون إلى برتقالي إضافةً لوضع إشارة عند مكان الخطأ.
2. إن إشارة * السابقة تدل على أن الملف لم يتم حفظه بعد.
3. Help File وهي عبارة عن تعليقات تشرح الهدف من الملف بهدف تنظيم العمل عند تطوير برامج أو خوارزميات معقدة، كما أنها تفيد المستخدم في العودة إلى ملفاته بسرعة كبيرة حيث يكتفي بقراءة هذه الملاحظات التي قام بكتابتها حتى يستطيع التعرف على عمل الملف، هذه التعليقات تأتي عادةً في السطور الأولى المكتوبة ضمن script.
4. Comments أي التعليقات، حيث يقوم المستخدم بتوضيح الهدف من تعريف متحول ما أو شرح عمل مجموعة من الأوامر والتعليمات مجتمعة بهدف فهم الملف بسرعة كبيرة.

   Comment thoroughly in the Script to avoid wasting time
5. Possible Breakpoints وهي عبارة عن الأماكن التي قام المستخدم بوضع Breakpoint فيها، الغرض من نقطة التوقف breakpoint هو نفسه في كافة لغات البرمجة وهو تحديد مواقع للتوقف ضمن برنامج ما بهدف اختبار وتصحيح ومعرفة قيم المتحولات أو خرج العمليات الحسابية المتعاقبة التي تتم ضمن البرنامج نفسه وخاصةً داخل حلقات for.
6. عدد أسطر الرماز المكتوب، يفيد ذلك عن حدوث خطأ في تحديد المكان بسرعة حيث أن MATLAB يعيد رسالة للمستخدم عن حدوث أي خطأ ويذكر به السطر التي تم وقوع الخطأ ضمنه.
7. Run Tools وهي أدوات متاحة من قبل محرر النصوص editor بعد تنفيذ الملف البرمجي، حيث يضاف إلى التنفيذ خيارات مختلفةً مثل تحديد زمن تنفيذ البرنامج المكتوب.

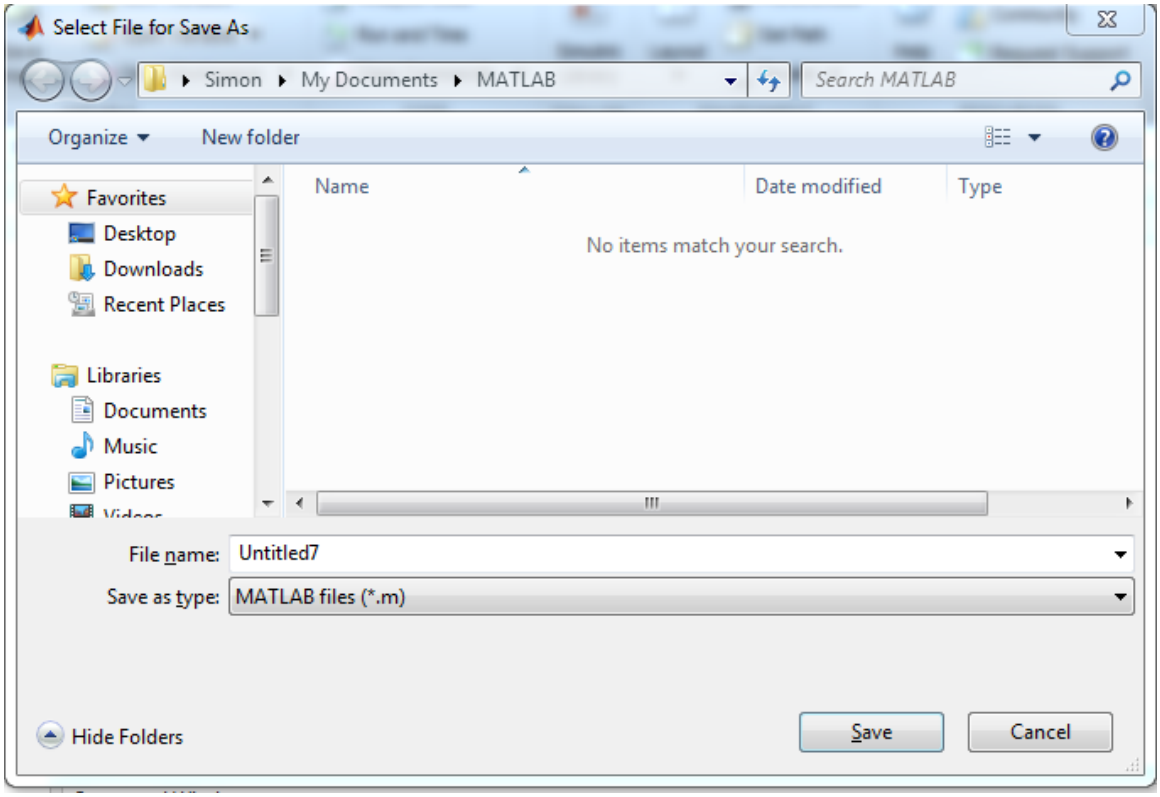
8. نقاط التوقف Breakpoints عند الضغط على الأيقونة السابقة يمكن اختيار وضع نقطة توقف أو إزالة نقاط توقف أو وضع نقاط توقف شرطية هذه الناقط تستخدم بشكل كبير عند التعامل ملفات script أو تنجيز التوابع.
9. Edit Tools يمكن عن طريق هذه الأزرار إضافة خيارات عديدة منها إضافة تعليق أو تقسيم الملف البرمجي إلى مقاطع section او ترتيب الرماز البرمجي المكتوب ضمن مستويات وهو ما يعرف ب Smart Indent.

كيف يتم حفظ ملف Script؟؟؟

قم بكتابة الرماز التالي ضمن Script:

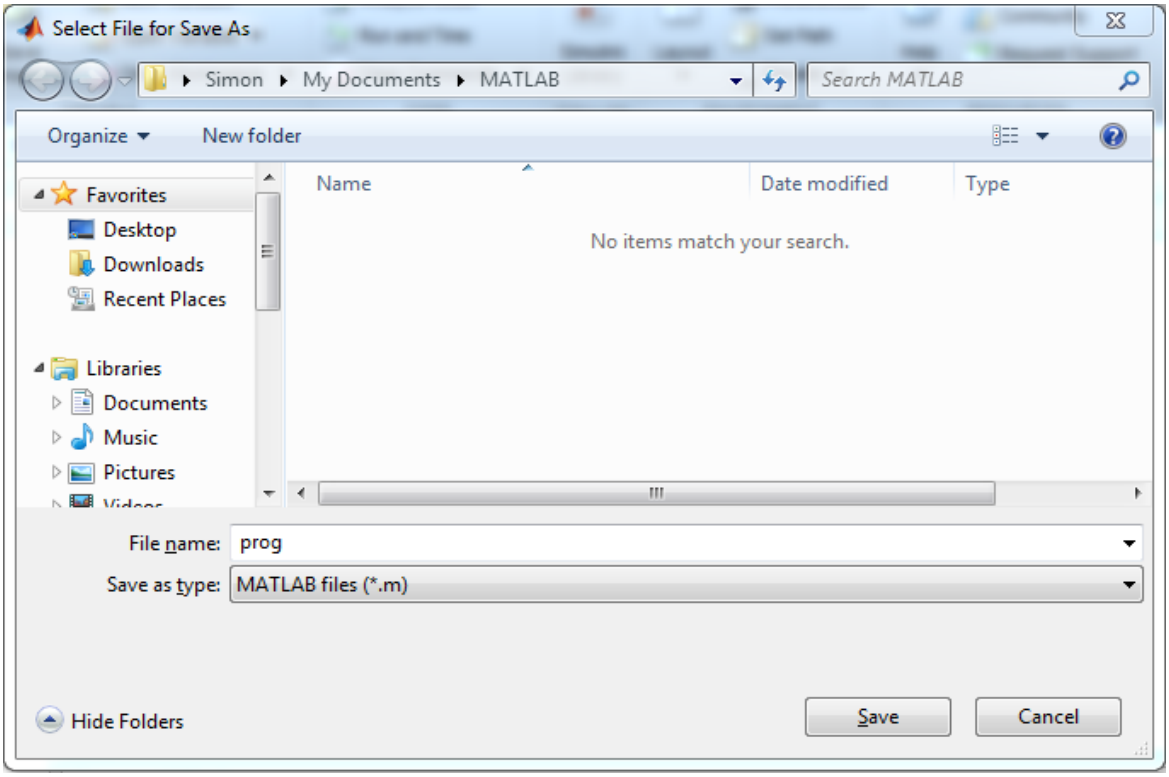
```
NumOfStudents=1000;
TeachStaff=15;
NonTeachStaff=2;
Total=NumOfStudents+TeachStaff+NonTeachStaff;
disp(Total);
```

قم بحفظ الملف التالي، اضغط Ctrl+S ستظهر النافذة التالية أمامك



ملاحظة: يمكن أيضاً حفظ الملف عن طريق عن طريق الضغط على زر Save ضمن Editor الموجود ضمن Toolstrip.

قم بتغيير اسم الملف من Untitled7 إلى prog واختر Save.

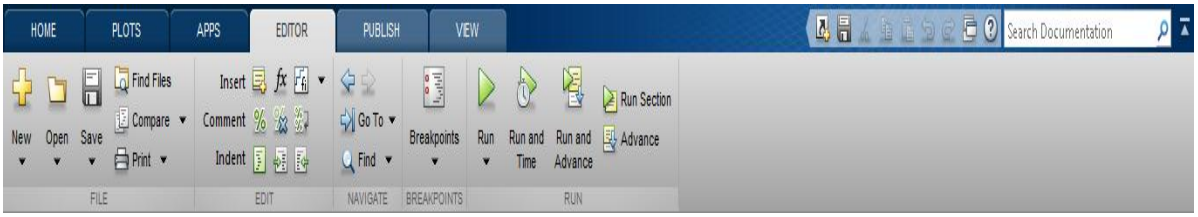


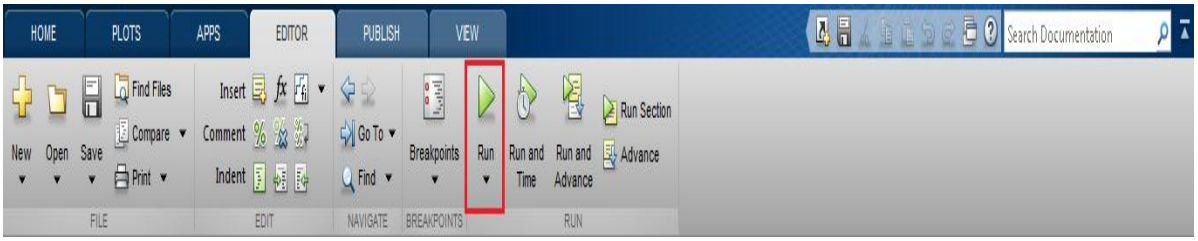
كيف يتم تنفيذ Run ملف Script؟؟؟

- عن طريق كتابة اسم الملف ضمن Command Window ثم الضغط على Enter.

مثلاً اكتب prog ضمن Command Window ستلاحظ تنفيذ الملف السابق وستظهر النتيجة ضمن Command Window كما هو موضّح في الشكل

- عن طريق الضغط على زر Run ضمن Editor الموجود ضمن Toolstrip.





لاحظ أنه عند تنفيذ أي Script يتم تعريف المتحولات الموجودة ضمنها في Work Space.

إضافة تعليقات إلى البرامج:

عند كتابة رماز برمجي باستخدام لغة MATLAB، فإن إضافة التعليقات Comments تعتبر عادة برمجية جيدة حيث تقوم هذه التعليقات بوصف الرماز البرمجي المكتوب، مما يتيح للشخص نفسه وللأشخاص الآخرين فهم الرماز المكتوب.

على غرار كافة الرمازات المكتوبة في MATLAB، يمكن لملفات Script أن تحتوي على تعليقات Comments ويمكن أيضاً للتوابع ان تحتوي عليها. يتم استخدام الرمز % (percent symbol) من أجل إضافة تعليقات، إن جميع الكلمات التي تتلو (تأبي بعد) إشارة % يتم التعامل معها على انها تعليقات أي لا يتم تنفيذها حتى وإن احتوت على تعليمات معروفة للغة MATLAB.

- يمكن للتعليقات ان تكون بنفس السطر الذي ننفذ/ نستدعي به التعليمة أو في سطر لاحق بهدف توضيح عمل التعليمات.

مثلاً:

```
% Add up all the vector elements.
y = sum(x)          % Use the sum function.
```

- التعليقات مفيدة في حالات تطوير برامج أو اختبارها، حيث يمكن منع بعض التعليمات من التنفيذ عن طريق جعلها كتعليق مثلاً عند إجراء اختبار على جزء محدد من البرنامج، يمكن استخدام }% و %{ من أجل جعل كتلة كاملة من التعليمات على أنها تعليق.

مثلاً:

```
a = magic(3);
%{
sum(a)
diag(a)
sum(diag(a))
%}
sum(diag(fliplr(a)))
```

إن المؤثر Operator " %{ و }% " الموثر هوة يلي ملون باللون الأزرق ، ملاحظة: من أجل الاستخدام الصحيح للمؤثر السابق يجب أن يوضع %{ في السطر السابق للكتلة المراد جعلها كتعليق و أن يوضع }% في السطر اللاحق للكتلة، ويجب ألا يحتوي السطر الذي يقع فيه هذا المؤثر على أي عبارة.

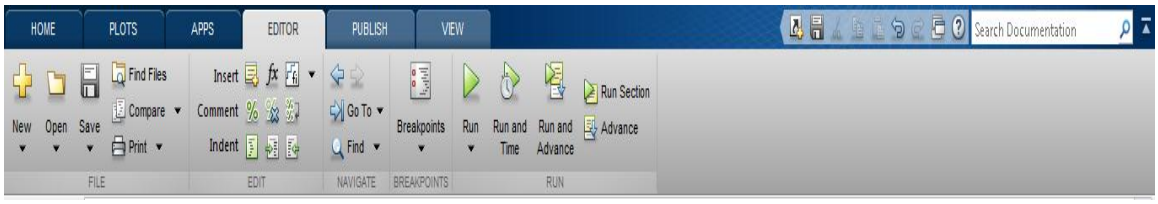
- من أجل إضافة تعليق لتعليمات تمتد على عدة سطور وتحتوي على المؤثر (...) يمكن استخدام المؤثر

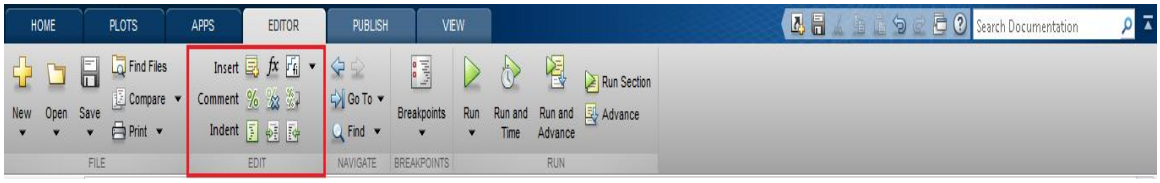
  (...) نفسه عوضاً عن استخدام %

مثلاً:

```
header = ['Last Name, ',      ...
          'First Name, ',     ...
      ... 'Middle Initial, ', ...
          'Title']
```

ضمن Editor يوجد عدد من الخيارات المفيدة لإضافة تعليقات في الجزء الموضح بالشكل





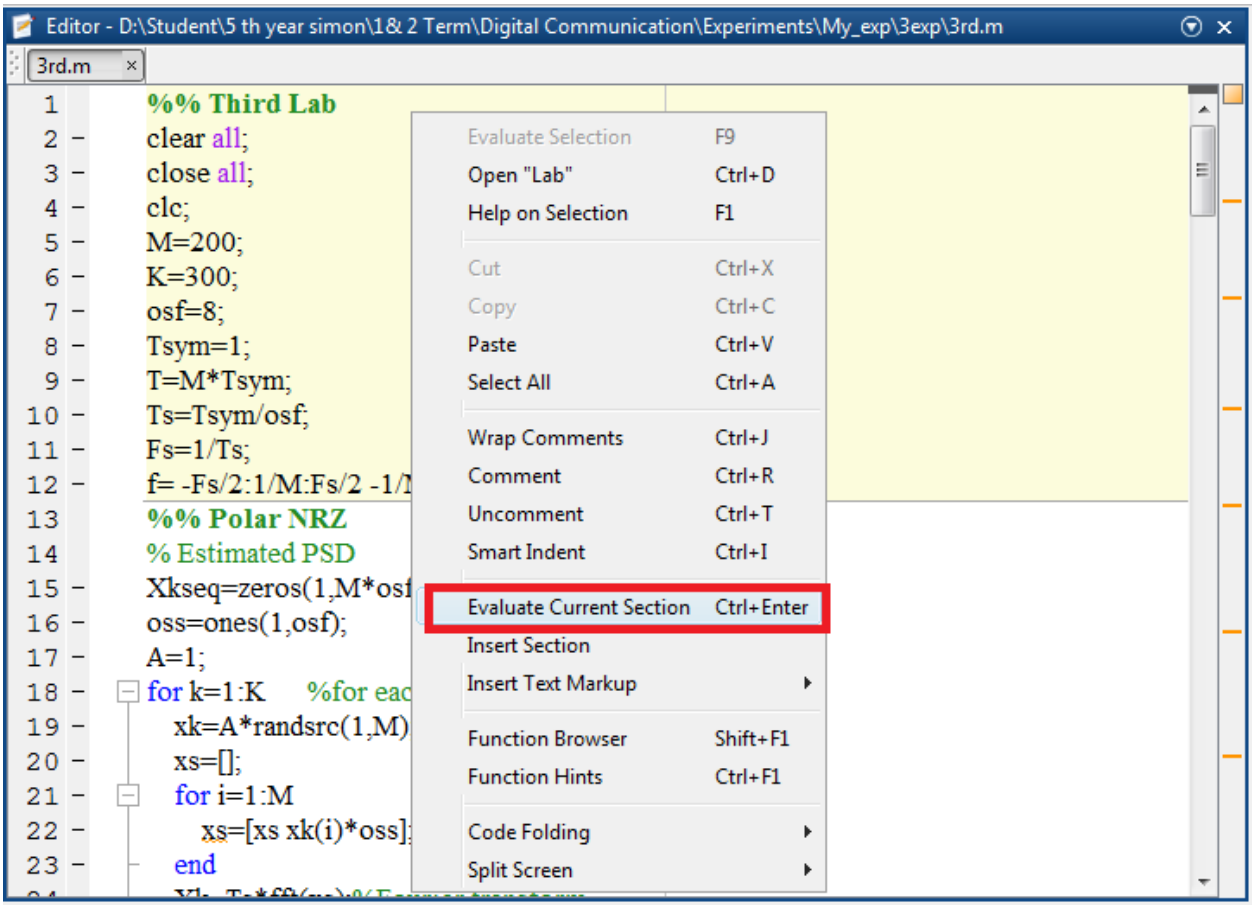
تنفيذ مقاطع من الرمازات (التقسيم إلى خلايا)

يتألف عادةً أي رماز برمجي مكتوب بلغة MATLAB أو أي ملف يحتوي على تعليمات من عدد كبير من التعليمات، وغالباً ما يهتم المبرمج بجزء من الرماز في لحظة ما من عمله ويحتاج مثلاً إلى تنفيذ مجموعة محدّدة من التعليمات دون الحاجة إلى تنفيذ باقي التعليمات لما قد يؤدي من مشاكل أو لما يتطلب من وقت عند تنفيذ البرنامج ككل، كذلك عند شرح الملفات أو الرمازات التي قمت بتنفيذها لشخص ما ستقوم بالشرح على شكل كتل كل منها له وظيفة محدّدة لذلك تعتبر عادة تجزئة الرماز البرمجي إلى عدة أقسام من أفضل العادات البرمجية وخاصةً عند استخدام لغة MATLAB.

من أجل تسهيل عملية تجزئة الرماز البرمجي إلى أقسام أو ما يعرف ضمن MATLAB بخلايا (code sections or code cells) يحتوي MATLAB على محارف خاصة تتيح لك تنفيذ مجموعة من التعليمات ضمن ملف script ما، هذه المحارف هي (%%).

عند تجزئة الرماز البرمجي إلى عدة أجزاء باستخدام %% يصبح بإمكاننا تنفيذ الخلية بشكل كامل عن طريق الضغط ضمن الخلية عندها سنلاحظ اختلاف لون الخلفية من الأبيض إلى الأصفر، ونضغط على الزر اليميني ونختار Evaluate Current Section أو نضغط Ctrl + Enter.

في الشكل التالي نلاحظ أنه في الجزء الأول قمنا بتعريف متحولات عامة ستفيدنا في باقي أجزاء العمل لذلك نضعها ضمن خلية وعندما نريد تعريف فقط المتحولات نقوم بتنفيذ التعليمات ضمن هذه الخلية ثم قمنا بتجزئة باقي الرماز إلى خلايا كل منها تقوم بوظيفة محدّدة.



كيف يتم إنشاء new function؟؟

سنعرض في هذه الفقرة تفاصيل إضافية عن التوابع ضمن MATLAB، لكن قبل الخوض في التفاصيل لابد لنا من توضيح الشكل العام لتعريف تابع وهو

Syntax of a function statement is:

Function [out1,out2,..., outN] = myfun (in1,in2,in3,..., inN)

أي يجب التصريح عن اسم التابع و دخله وخرجه.

- إن التعريف السابق Function [out1,out2,...,outN] = myfun (in1,in2,in3,...,imN) يصرّح عن تابع باسم myfun والذي يقبل in1,in2,in3,...,imN كمتحولات دخل له، ويعطي خرج out1,out2,..., outN، إن التعريف السابق يجب وضعه في أول الملف عند تعريف تابع، أما للتعبير عن نهاية التابع يمكن استخدام "end" في نهاية ملف التابع.
- اللاحقة التي يتم حفظ الملف فيها هي .m، من أجل أن يكون اسم التابع صحيحاً valid يجب أن يبدأ بحروف أبجدية، يمكن أن يحتوي اسم التابع (وليس أول حرف) أحرف وأرقام و "_" underscores.
- يمكن للتوابع أن تقبل أكثر من دخل يتم التصريح عنه كمايلي:

```
function y = myfunction(one,two,three)
```

- يمكن للتوابع أن تعيد أكثر من خرج يتم التصريح عنه كمايلي:

```
function [one,two,three] = myfunction(x)
```

- يكون للتابع ألا يعيد أي خرج ويتم التصريح عنه كما يلي:

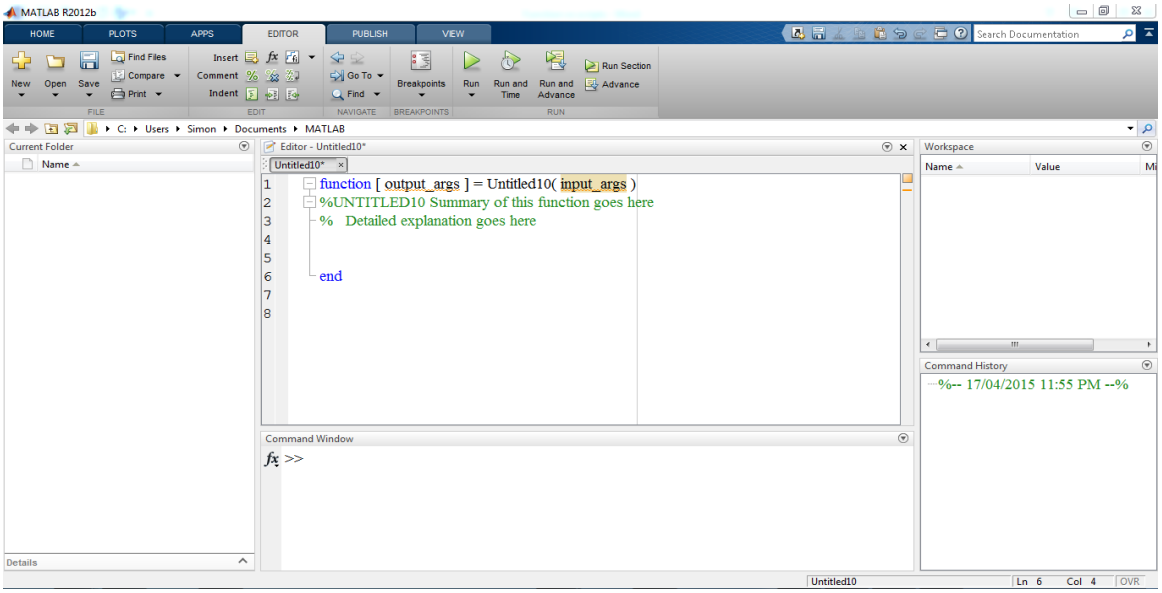
```
function myfunction(x)
```

```
function [] = myfunction(x)
```

- يكون اسم التابع مطابقاً لاسم الملف التي يتم حفظه.
- بالنسبة للمتحولات التي يتم تعريفها داخل التابع و لكن لا يتم إعادتها من قبل التابع (أي أنها ليست خرج للتابع) تختفي عندما يتوقف تشغيل البرنامج، على الرغم من أنه يتم إنشاؤها وإسناد القيم لها، هذا الأمر يرجع إلى وجود فضاء عمل Workspace يختلف عن Workspace الخاص بالبرنامج، ضمن هذا الفضاء يتم تعريف المتحولات الخاصة بالتوابع دون السماح للوصول إليها إلا في حالة وضه نقاط توقف Break points، ونذكر بأن فضاء العمل Workspace الخاص بالبرنامج يسمى عادةً Base Workspace
- كما ذكرنا في الفقرات السابقة حول التعليقات يمكن إضافة تعليقات لمفلات التوابع أيضاً بشكل مشابة لإضافتها لملفات Scripts.

من أجل إنشاء تابع جديد، من خلال شريط Toolstrip ثم نافذة Home قم بالضغط على الزر  ثم new اختر function

عند إنشاء تابع جديد ستصبح الواجهة الأساسية ل MATLAB على الشكل التالي.



نلاحظ إنشاء ملف جديد ولكن لا يكون فارغاً blank كما الحال عند إنشاء script جديدة، إنما يحتوي على الترويسة الأساسية التي تحتوي على المعلومات الأساسية للتصريح عن تابع.

ملاحظة: يتم حفظ التابع بنفس الطريقة التي يتم حفظ ملفات Script. كما يجب التذكير **بأن يكون اسم التابع مطابقاً لاسم الملف التي يتم حفظه.**

أمثلة عن إنشاء توابع:

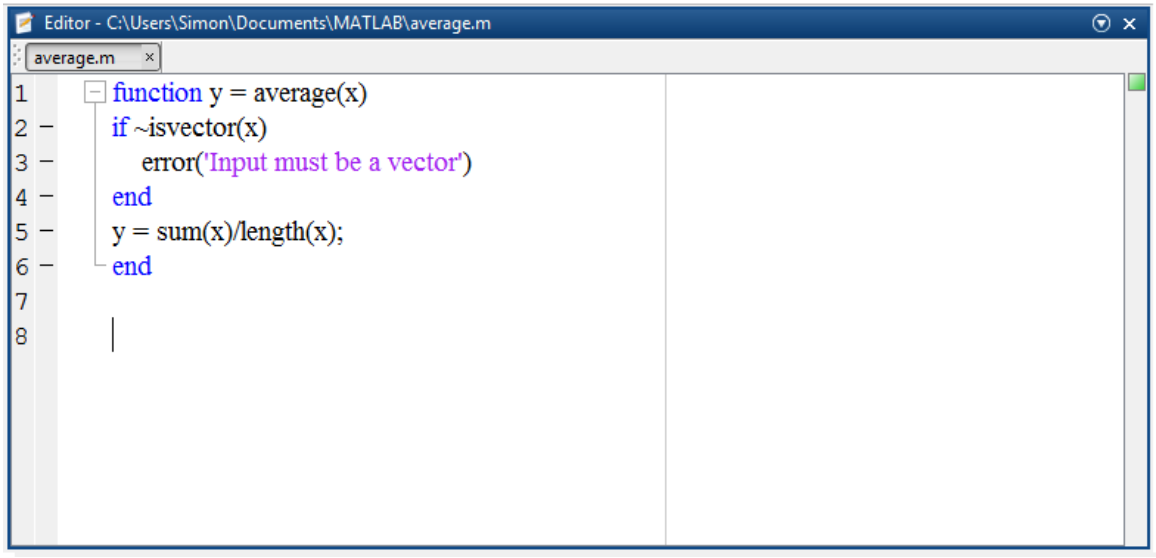
مثال 1:

قم بتعريف تابع في ملف يدعى average، سيتم حفظه على الشكل average.m، يقوم هذا التابع بأخذ شعاع vector كدخل له، ثم يقوم بحساب متوسط القيم الموجودة ضمن الشعاع، ويعيد كخرج له قيمة وحيدة هي قيمة المتوسط.

التابع:

```
function y = average(x)
if ~isvector(x)
    error('Input must be a vector')
end
y = sum(x)/length(x);
end
```

ثم قم بحفظه باسم average، صورة للتابع ضمن بيئة MATLAB.



قم ضمن Command window بإنشاء شعاع يحتوي على القيم من 1 إلى 99 واسمه z ثم قم استدعاء التابع كما يلي، ستحصل على النتيجة التي تعبر عن المتوسط وهي 50.

ملاحظة: يمكن إنشاء script وتعريف الشعاع ضمنها واستدعاء التابع.

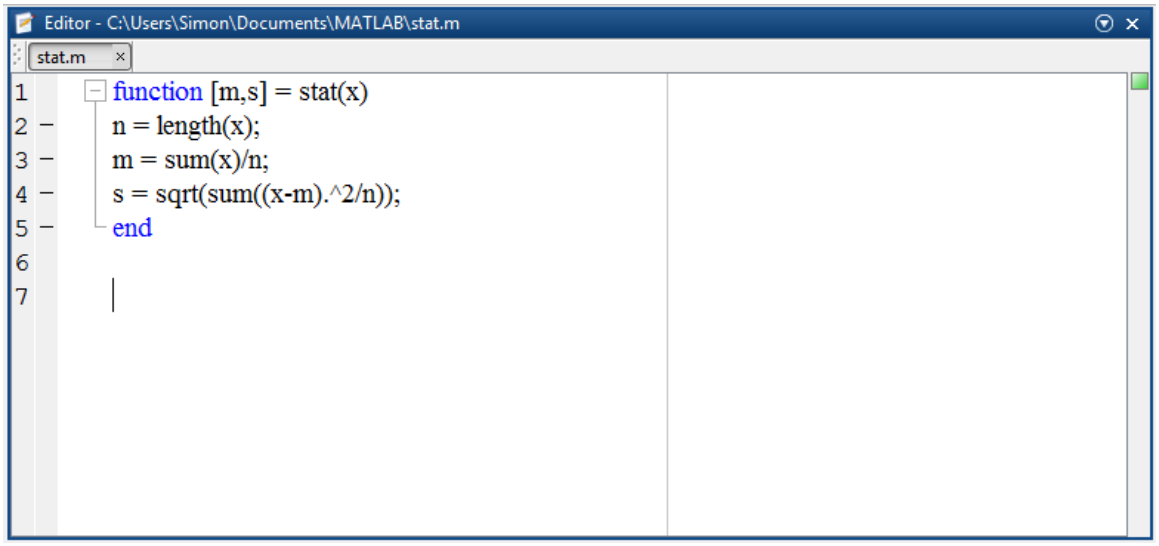
مثال 2:

قم بتعريف تابع في ملف يدعى stat، يقوم بإعادة المتوسط والانحرف المعياري لشعاع يمثل الدخل.

التابع:

```
function [m,s] = stat(x)
n = length(x);
m = sum(x)/n;
s = sqrt(sum((x-m).^2/n));
end
```

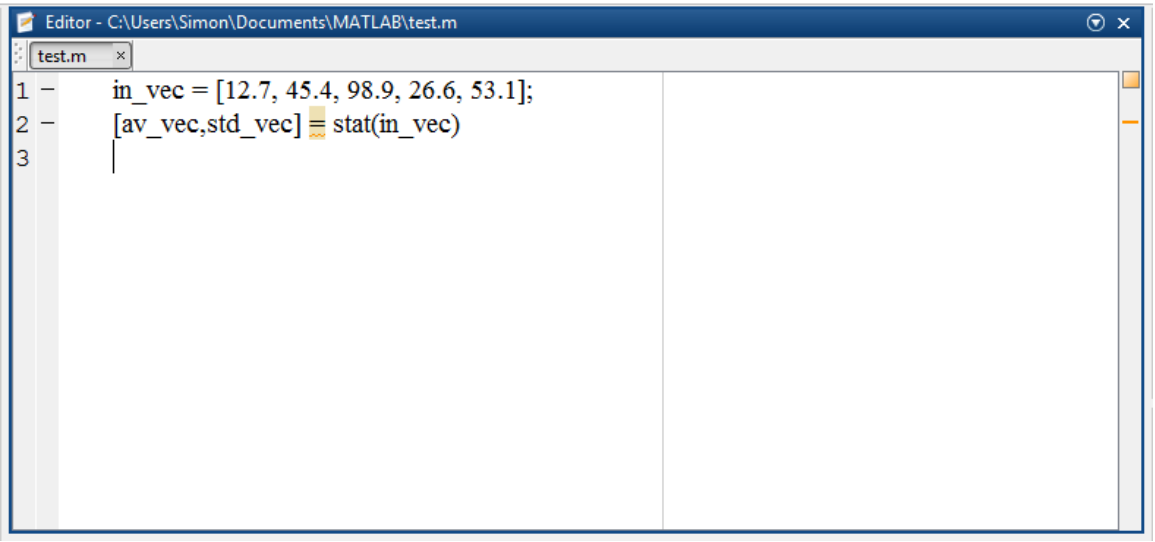
ثم قم بحفظه باسم stat، صورة للتابع ضمن بيئة MATLAB.



قم بإنشاء ملف script جديد، الغرض منه هو إجراء اختبار للتابع السابق، قم بتسمية الملف بـ test، ثم عرّف قيم شعاع الدخل، وقم باستدعاء التابع وفقاً للرماز التالي:

```
in_vec = [12.7, 45.4, 98.9, 26.6, 53.1];
[av_vec,std_vec] = stat(in_vec)
```

صورة لscript ضمن بيئة MATLAB.



عند تنفيذ التعليمات السابقة، عن طريق تحديدها ثم الضغط على الزر اليميني واختيار "Evaluate Selection" أو الضغط على F9، نحصل ضمن Command Window على الخرج التالي، أو يمكن قراءة قيمة كل متحول ضمن work space.

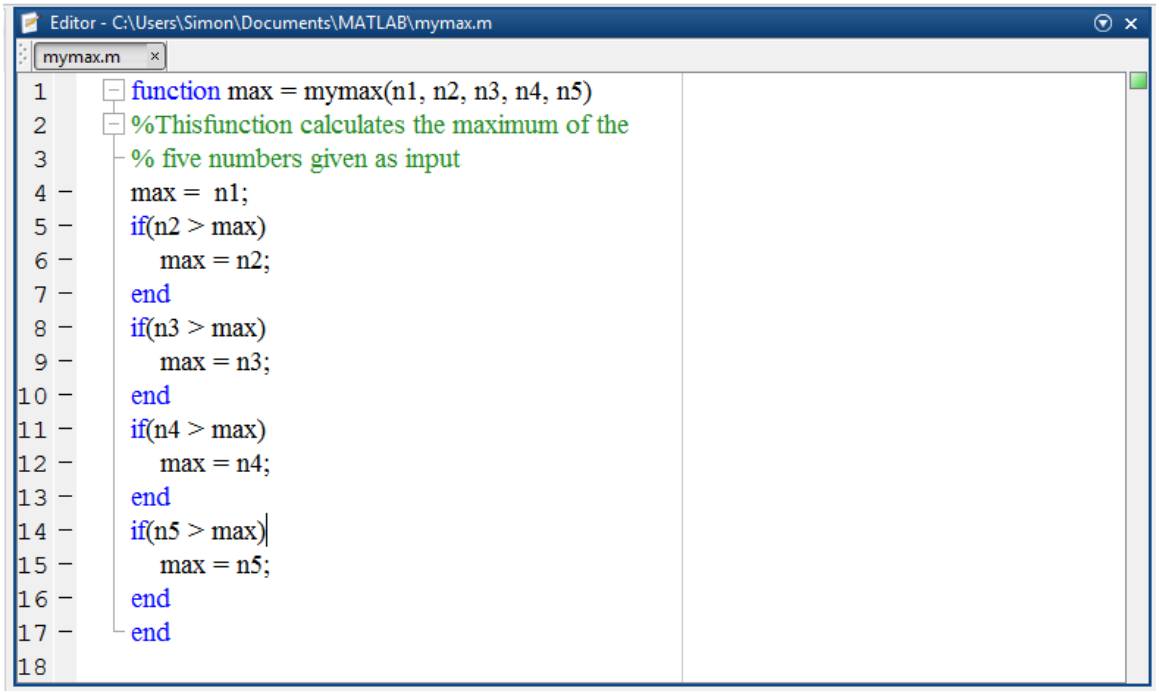
مثال 3:

قم بتعريف تابع في ملف يدعى mymax، يأخذ خمسة متحولات دخل عبارة عن أرقام ثم يقوم المقارنة قيما بينها من أجل أيجاد العدد الأكير فيما بينها، حيث أن الخرج هو القيمة الأكبر ضمن عناصر الدخل

التابع:

```
function max = mymax(n1, n2, n3, n4, n5)
%Thisfunction calculates the maximum of the
% five numbers given as input
max =  n1;
if(n2 > max)
    max = n2;
end
if(n3 > max)
    max = n3;
end
if(n4 > max)
    max = n4;
end
if(n5 > max)
    max = n5;
end
end
```

ثم قم بحفظه باسم mymax، صورة للتابع ضمن بيئة MATLAB.



يمكن إضافة التعليقات ضمن نص التابع وذلك لتوضيح عمل التابع، يجري ذلك عن طريق استخدام "%"، إن إضافة التعليقات أمر ضروري عند تنفيذ أي عمل حيث تفيد في توضيح معنى كل متحول، توضيح الهدف من كل تعليمة أو مجموعة من التعليمات، كما أنّه عند العودة إلى التابع يكفي قراءة التعليقات الموجودة من أجل معرفة عمل.

ضمن التابع السابق تم كتابة التعليقات التالية:

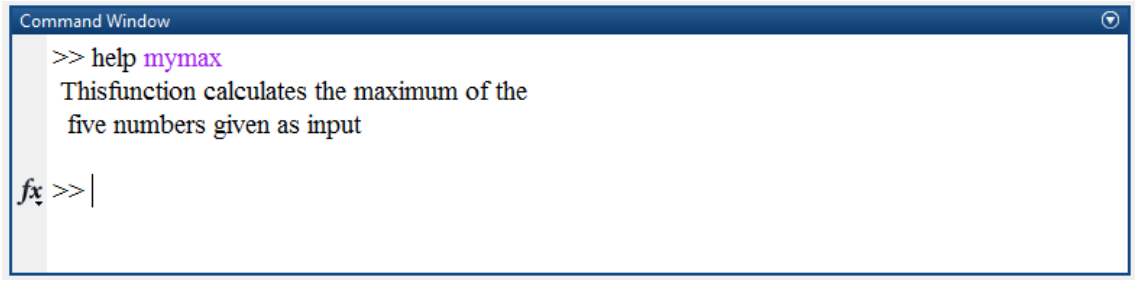
%Thisfunction calculates the maximum of the

% five numbers given as input

إن التعليقات التي تأتي مباشرةً بعد التصريح عن التابع، تؤمن مساعدة حول التابع عند الاستفسار عن عمل التابع، اكتب ضمن Command Window التعليمة التالية:

help mymax

سيكون خرج هذه التعليمة



ضمن Script التي تم إنشاؤها مسبقاً، قم باستدعاء التابع على الشكل التالي:

mymax(34,78,89,23,11)

سيكون الخرج على الشكل التالي:

```
ans=
89
```

تحويل Script إلى Function:

قم بإنشاء ملف Script جديد ثم قم بكتابة التعليمات التالية، إضافةً لذلك قم بحفظ الملف وتسميته باسم triarea. الهدف من هذه التعليمات هو حساب مساحة مثلث

```
b = 5;
h = 3;
a = 0.5*(b.* h)
```

يمكن ضمن Command Window استدعاء التابع عن طريق كتابة اسمه ثم اضغط Enter، أي قم بكتابة triarea

سيكون الخرج على الشكل:

ذكرنا أن التوابع مفيدة أكثر من ملفات Script يكونها أكثر عمومية، لنوضح الفكرة السابقة من خلال المثال التالي:

إن ملف triarea التي تم كتابته والذي يفيد في حساب مساحة مثلث يتم تطبيقه على القيم السابقة فقط، اما في حال حساب مساحة مثلث جديد باستخدام script السابقة، يمكن إعادة إسناد القيم للمتحولين b و h المعبران عن طول القاعدة والارتفاع على الترتيب، ومن ثم تنفيذ ملف script مرة أخرى وهكذا، وفي كل مرة يتم تخزين النتيجة في المتحول a ضمن workspace.

يمكن تحويل script السابقة إلى تابع أكثر عمومية ومرونةً بحيث لا نحتاج كل مرة إلى تغيير قيم المتحولات ضمن script بشكل يدوي، بل يكفي أن نمرر هذه المتحولات كدخل للتابع ومن ثم سيقوم بحساب المساحة.

رماز التابع

```
function a = mytriarea(b,h)
a = 0.5*(b.* h);
```

بعد حفظ التابع، يمكن استخدام التابع من أجل عدة قيم لطول القاعدة والارتفاع ضمن Command Window دون الحاجة للتعديل على ملف script، على سبيل المثال:

```
a1 = mytriarea(1,5)
a2 = mytriarea(2,10)
a3 = mytriarea(3,6)
```

سيكون الخرج على الشكل التالي:

```
a1 =
    2.5000
a2 =
    10
a3 =
     9
```

أنواع التوابع Function Types:

يوجد ضمن MATLAB عدد كبير من أنواع التوابع مثل Local function, Nested function, Private function and Anonymous function سنتطرق للحديث عنها ويمكن القراءة عنها ضمن help بهدف زيادة المعرفة.

1. Anonymous Functions التوابع مجهولة الاسم:

- يشابه anonymous function في عمله أي تابع inline function في لغات البرمجة التقليدية.
- يتم تعريف مثل هذا النوع من التوابع باستخدام تعليمة وحيدة في لغة MATLAB، ويتألف من تعبير رياضي mathematical expression وحيد و عدد اختياري من متحولات الدخل والخرج.
- يمكن تعريف تابع من نمط anonymous ضمن Command Window أو ضمن Script أو ضمن تابع آخر.

من أجل تعريف تابع من نمط anonymous يجب اتباع الشكل التالي للتعريف

The syntax is

```
f =@(arglist)expression
```

مثال:

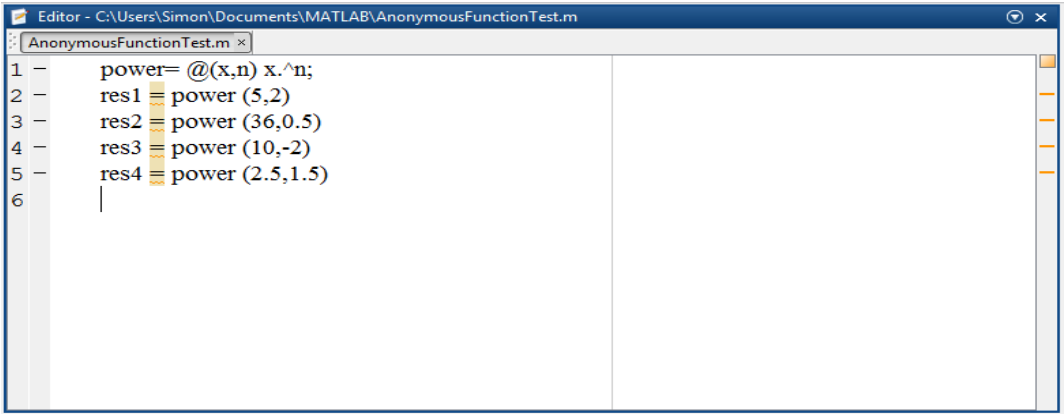
في هذا المثال سنقوم بكتابة تابع من نمط anonymous يدعى power ضمن script باسم AnonymousFunctionTest، والذي يأخذ متحولين كدخل له، ويقوم باعتبار العدد الأول كأساس للقوة ويرفع الأساس إلى قوة العدد الثاني، ثم قم باختبار التابع من أجل عدة حالات وهي:

رفع العدد 5 إلى القوة 2 أي إيجاد تربيع القيمة، رفع العدد 36 إلى القوة 0.5 أي إيجاد جذر القيمة، رفع العدد 10 إلى القوة 2-، رفع العدد 2.5 إلى القوة 1.5 .

الحل:

```
power= @(x,n) x.^n;
res1 = power (5,2)
res2 = power (36,0.5)
res3 = power (10,-2)
res4 = power (2.5,1.5)
```

يوضح الشكل ملف script بعد تعريف التابع من نمط Anonymous و إضافةً للأمثلة السابقة



الخرج

- ملاحظة: يعتبر هذا النمط من التوابع هو الوحيد القادر على إنشاء توابع بسيطة دون الحاجة إلى إنشاء ملفات خاصة لهم، وهذه الخاصية مفيدة حيث أن كثرة الملفات أحياناً قد تتسبب ببعض المشاكل مثل ضياعها مثلاً، أو حذف أحدها مما يعطل عمل خوارزمية بأكملها !!!

مثال:

سنقوم بإنشاء تابع يقوم بحساب العلاقة التالية:

$$y = a\,x^2 + bx + c$$

ولنختر قيم الأمثال كالتالي:

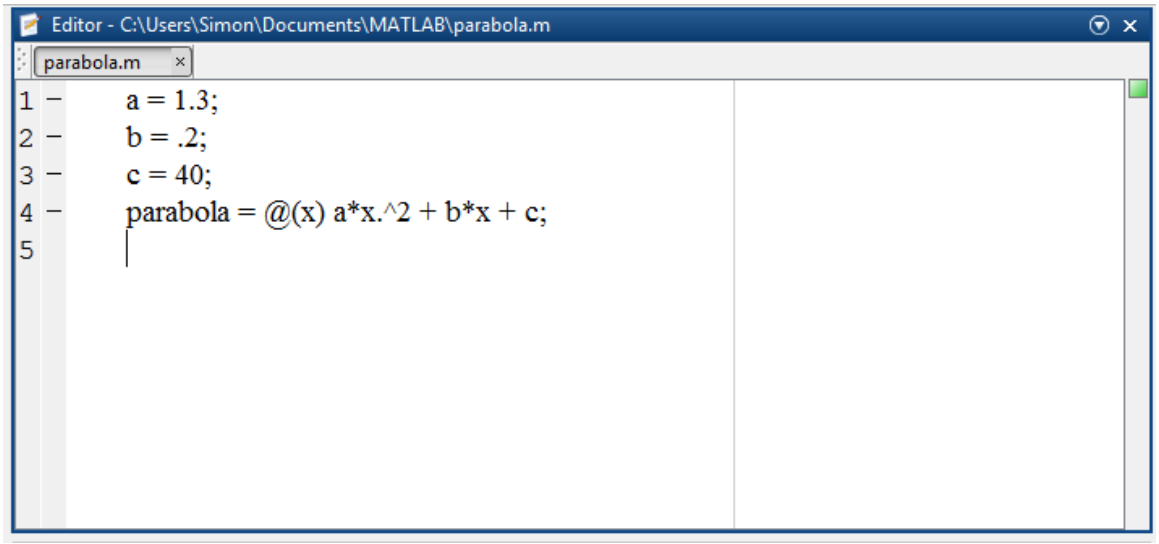
a=1.3, b=0.2, c=30.

سنوضح في هذا المثال أن التابع المعرف من النمط Anonymous لا يقوم فقط بحفظ التعبير expression المراد حسابه إنما يقوم أيضاً بحفظ المتحولات التي يحتاج لها التعبير من أجل إيجاد القيمة الحسابية.

الحل:

أولا قم بتعريف المتحولات التالية، إضافةً للتابع من نمط Anonymous المطلوب.

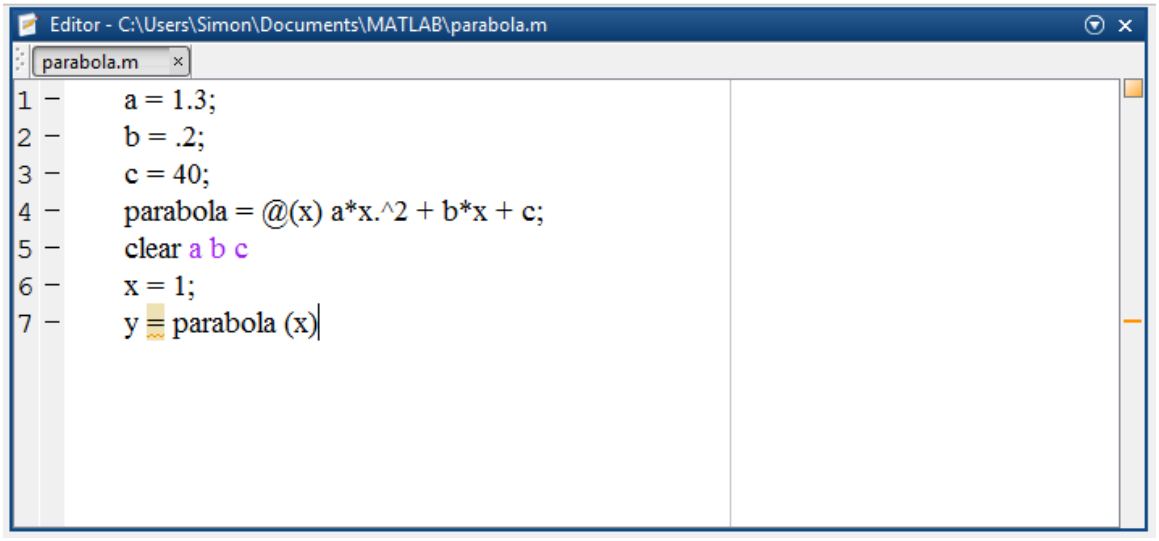
```
a = 1.3;
b = .2;
c = 40;
parabola = @(x) a*x.^2 + b*x + c;
```



ثم قم بتحديد التعليمات السابقة واضغط الزر اليميني واختار evaluate selection أو اضغط F9

ثانياً قم بسمح المتحولات a , b , c عن طريق التعليمة clear ثم استدعي التابع من أجل القيمة x = 1 وقم بإسنادها للمتحول y.

```
clear a b c
x = 1;
y = parabola (x)
```



الخرج

2. Local Functions التوابع المحلية:

- ضمن MATLAB أيّ نمط للتوابع باستثناء نمط anonymous يجب تعريفه ضمن ملف منفصل، حيث يحتوي كل ملف خاص بالتوابع (بلاحقة .m) على تابع رئيسي primary function أو main function يتم برمجته أولاً ضمن الملف ومن ثم **يمكن** أن يحتوى على أي عدد من التوابع الثانوية/الفرعية sub-function ،وهي اختيارية أي ليس من الضروري وجودها على عكس primary function، حيث يتم تعريف هذه التوابع الثانوية الاختيارية بعد تعريف التابع الرئيسي ويتم استخدامها من قبله. تفيد مثل هذه التوابع في تقسيم البرنامج إلى مهمات tasks أصغر مما يسرع من زمن التنفيذ، ويسهل فهم البرنامج.
- لا يمكن استدعاء التوابع الثانوية عن طريق command window او التوابع الاخرى خارج الملف الموجودة ضمنه هذه التوابع الثانوية، على عكس التوابع الرئيسية حيث يمكن استدعائها عن طريق command window او التوابع الاخرى خارج الملف الموجودة ضمنه.
- Local function التوابع المحلية هي عبارة عن توابع متاحة لأي توابع أخرى موجودة معها ضمن نفس الملف، ترد هذه التوابع بعد التابع الرئيسي عادةً.
- ملاحظة: جميع التوابع وبما فيهم التوابع المحلية لديهم فضاء العمل workspace الخاص بهم وهو منفصل عن فضاء العمل الأساسي في البرمجية والذي يسمى عادةً base workspace.

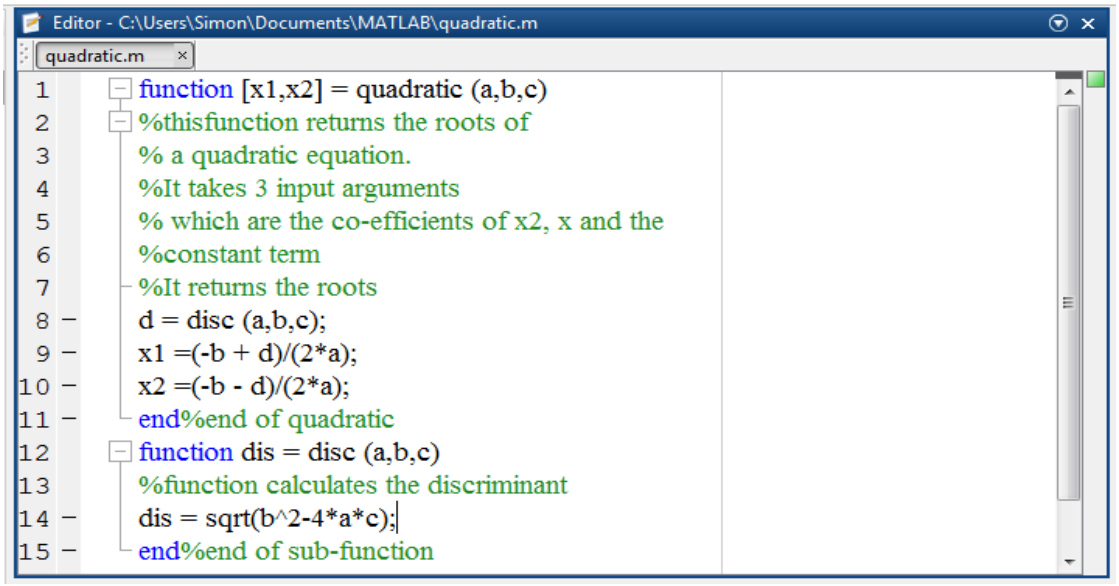
مثال:

لنقم بكتابة تابع يسمى quadratic والذي يقوم بحساب جذور معادلة تربيعية، يجب أن يمرر للتابع ثلاث متحولات دخل تعبر عن الأمثال للحد التربيعي، والحد الخطي، والحد الثابت. يعيد التابع متحولي خرج يعبران عن الجذور.

يحتوي ملف التابع quadratic.m على تابع رئيسي هو quadratic وتابع محلي هو disc والذي يقوم بحساب المميّز دلتا. قم بإنشاء ملف تابع جديد وتسميته باسم quadratic واكتب الرماز التالي:

```
function [x1,x2] = quadratic (a,b,c)
%thisfunction returns the roots of
```

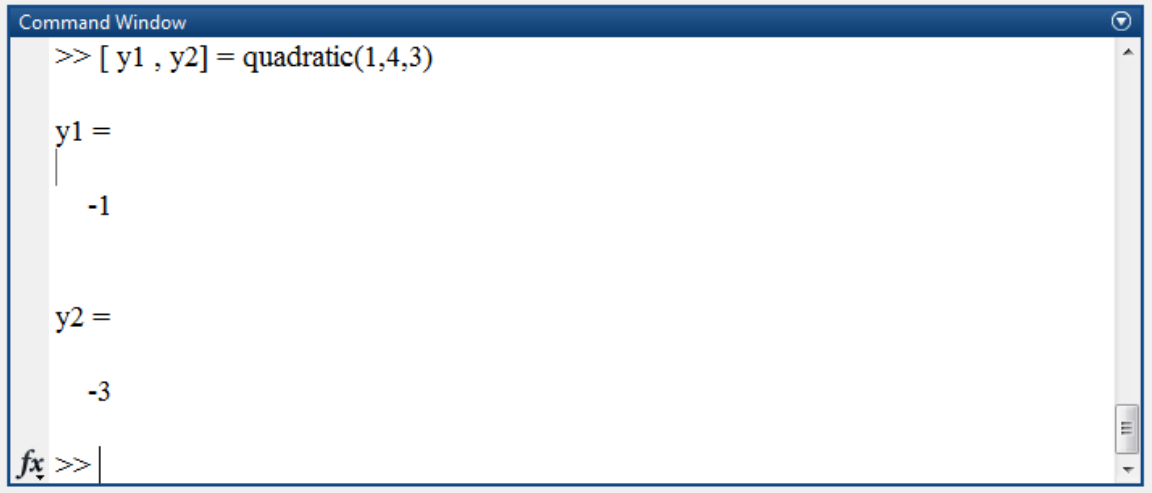
```
% a quadratic equation.
%It takes 3 input arguments
% which are the co-efficients of x2, x and the
%constant term
%It returns the roots
d = disc (a,b,c);
x1 =(-b + d)/(2*a);
x2 =(-b - d)/(2*a);
end%end of quadratic
function dis = disc (a,b,c)
%function calculates the discriminant

dis = sqrt(b^2-4*a*c);
end%end of sub-function
```



يمكن استدعاء التابع من خلال command Window مثلاً اكتب:

```
[ y1 , y2] = quadratic(1,4,3)
```

فنحصل على الخرج التالي:



مثال:

التابع التالي

```
function b = myfunction(a)
   b = squareMe(a) + doubleMe(a);
end
function y = squareMe(x)
   y = x.^2;
end

function y = doubleMe(x)

   y = x.*2;
end
```

يقوم بحساب مجموع قيمة مربع الدخل وحاصل ضرب الدخل بـ 2. نلاحظ ضمن التابع وجود تابعين محليّين هما squareMe و doubleMe.

3. Nested Functions التوابع المتداخلة:

يمكن تعريف توابع داخل ضمن جسم تابع آخر، هذه التوابع تعرف باسم التوابع المتداخلة nested functions، يتم تعريف التابع من نمط nested تماماً كما يتم تعريف أي تابع آخر. والاختلاف الرئيسي بين التوابع المتداخلة والتوابع المحلية هو ان التوابع المحلية يمكن أن تستخدم أو تعدل متحولات معرّفة في التابع الرئيسي دون أن يتم تمرير هذه المتحولات صراحةً للتابع المتداخل كمتحول دخل.

يتبع تابع من نمط nested النحو syntax التالي:

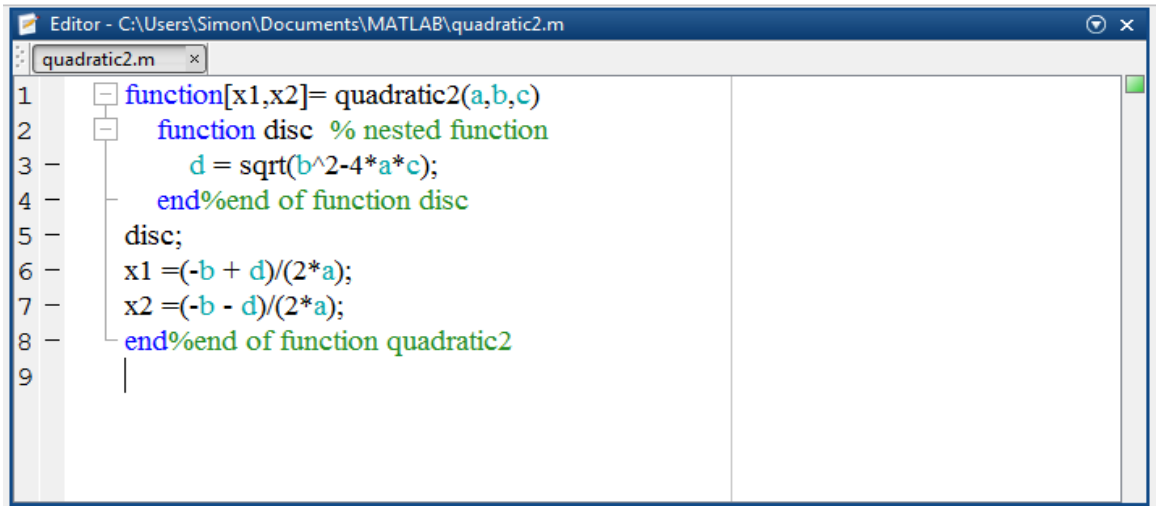
```
function x = A(p1, p2)

...
```

```
    B(p2)
  function y = B(p3)
    ...
  end
...
end
```

مثال:

سنعيد كتابة التابع السابق quadratic ولكن في هذه المرة سنجعل التابع disc عبارة عن تابع من نمط nested وليس تابع من نمط local.

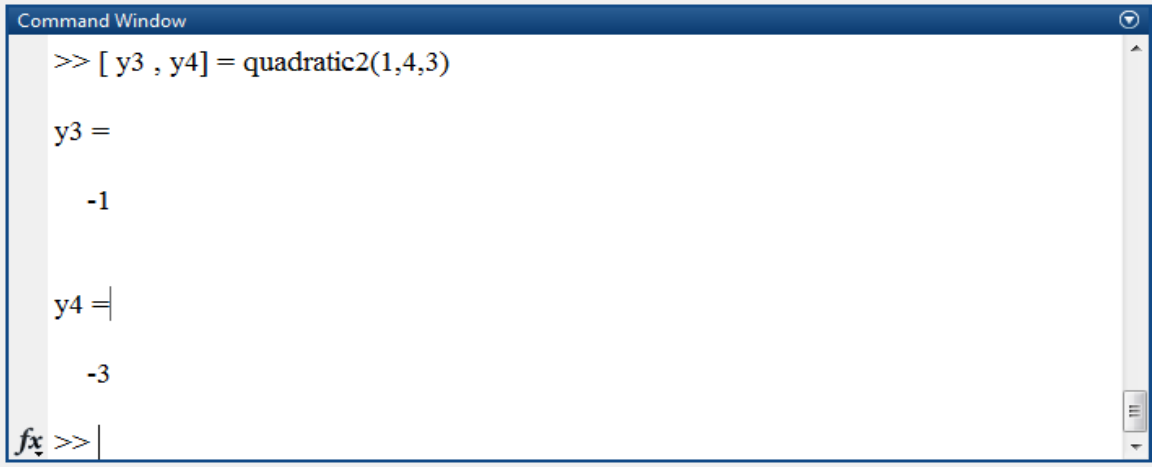
```
function[x1,x2]= quadratic2(a,b,c)
  function disc  % nested function
      d = sqrt(b^2-4*a*c);
  end%end of function disc
disc;
x1 =(-b + d)/(2*a);
x2 =(-b - d)/(2*a);
end%end of function quadratic2
```



يمكن استدعاء التابع من خلال command Window مثلاً اكتب:

```
[ y3 , y4] = quadratic2(1,4,3)
```

فنحصل على الخرج التالي:

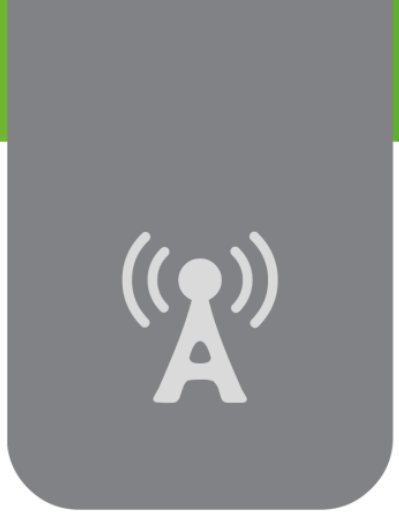


لن نتطرق للحديث عن private function حيث لا نحتاج لها ضمن مقررنا هذا، للمزيد من المعلومات حولها استخدم help ضمن MATLAB.

ملاحظة: يوجد عدد من الأمور التي لابد من الاطلاع عليها تجدونها ضمن MATLAB HELP وفق المسار MATLAB → Programming Scripts and Functions → Debugging → Examples and How To → Ways to Degug MATLAB Files … …. Error Brealpoints

يفضل قراءة كافة الفقرات حيث عددها 11 فقرة ضمن صفحة وحيدة تفتح عند الضغط على أي من العناوين.

https://www.youtube.com/watch?v=pRsGM7H91VY



الفصل الرابع: البرمجة في MATLAB® -2- (عبارات التحكم بالتدفق، البيانيات والإظهار، السلاسل المحرفية)

Programming in MATLAB® -2- (Control Flow Statements, Graphics and Visualization, Strings)

عنوان الموضوع:

البرمجة في MATLAB® -2- (عبارات التحكم بالتدفق، البيانيات والإظهار، السلاسل المحرفية)

Programming in MATLAB® -2- (Control Flow Statements, Graphics and Visualization, Strings)

الكلمات المفتاحية:

MATLAB، تعابير التحكم بالتدفق Control Flow Statements، العبارات الشرطية Conditional Statements، عبارة if statement if، عبارات if متداخلة nested if ، عبارة switch switch statement، الحلقات Loops، حلقة while loop while، حلقة for loop for، السلاسل المحرفية Strings ، المحارف Chars، دمج concatenation، البيانيات Graphics، رسم plot.

ملخص:

نعرض في هذا الفصل عدد من المفاهيم البرمجية الهامة التي تساعد المستخدم في بناء رمازات وبرامج فعّالة ومفيدة. سنتعرف في الجزء الأول من هذا الفصل على عبارات التحكم بالتدفق وخصوصاً العبارات الشرطية، مثل عبارة if و switch، التي تساعد المستخدم على اتخاذ القرار مثلاً وفقاً لقيم الدخل أو استثناء الحالات الشاذة، ثم ننتقل إلى الحلقات والتي تعتبر جزءاً أساسياً ضمن MATLAB وخاصةً الشرطية منها.

نقدم في الجزء الثاني من الفصل خيارات إضافية للتعامل مع السلاسل المحرفية، وأخيراً ننتقل إلى موضوع هام جداً وهو إظهار المتحولات، التوابع والنتائج فبعد القيام بأية عملية حسابية أو تطبيق أية خوارزمية لابد لنا من رسم النتائج إما بهدف فهم الظاهرة، أو مناقشة النتائج أو عرضها بشكل واضح. سيتم التعرف على الخيارات والتوابع العديدة المتوفرة ضمن برمجية MATLAB التي تساعد في عملية الإظهار.

أهداف تعليمية:

يتعرف الطالب في هذا الفصل على:

- استخدام تعابير التحكم بالتدفق.
- إتقان العبارات الشرطية و الحلقات ضمن برمجية MATLAB.
- التعامل مع السلاسل المحرفية، وإتقان تطبيق العمليات عليها.
- استخدام توابع MATLAB في عملية الإظهار.
- استخدام وإتقان المعارف ضمن هذا الفصل وماكملتها مع الفصول السابقة لتحليل وتصميم رمازات وبرامج متقدمة تعطي حلاً لمشكلة معينة.

تعابير التحكم بالتدفق Control Flow Statements

وجدنا في الفصول السابقة أن التعليمات Commands والتعابير Expressions ضمن MATLAB يجري تنفيذها بشكل تسلسلي، ولكن الحال لا يكون دائماً كما سبق، ففي كثير من الأحيان واعتماداً على معطيات الدخل، أو خرج مرحلة ما من خوارزمية، يتم تنفيذ مجموعة محدّدة من التعليمات وذلك باستخدام العبارات الشرطية Conditional Statements أو غيرها من عبارات التحكم بالتدفق.

يشتمل MATLAB على عدد كبير من التعابير المستخدمة في التحكم بتدفق المعطيات كالتعابير الشرطية، إضافةً لماسبق يحتوي MATLAB على عبارت تسمح بتكرار مجموعة من التعليمات وهو ما يعرف بالحلقات Loop.

نعرض فيما يلي العبارات (أو التعابير) الخاصة بـ Control Flow ضمن MATLAB.

التعليمة Command	الوصف Description
if/elseif/else	تنفيذ مجموعة من العبارات إزا كان الشرط محقق (صحيح true).
for	تنفيذ مجموعة من العبارات عدد محدّد من المرات.
switch/case/otherwise	التبديل بين عدة حالات cases اعتماداً على تعبير ما.
try/catch	تنفيذ مجموعة من العبارات وتلتقط الأخطاء الناتجة.
while	تنفيذ مجموعة من العبارات بشكل متكرر **طالما** أن الشرط محقق (صحيح true).
break	إنهاء تنفيذ حلقة for أو while.
continue	تمرير التحكم إلى التكرار التالي next iteration لحلقة for أو while.
end	إنهاء مجموعة من التعليمات أو الإشارة لنهاية فهرسة صفيفة.
pause	توقيف التنفيذ مؤقتاً.
return	العودة إلى استدعاء التابع.

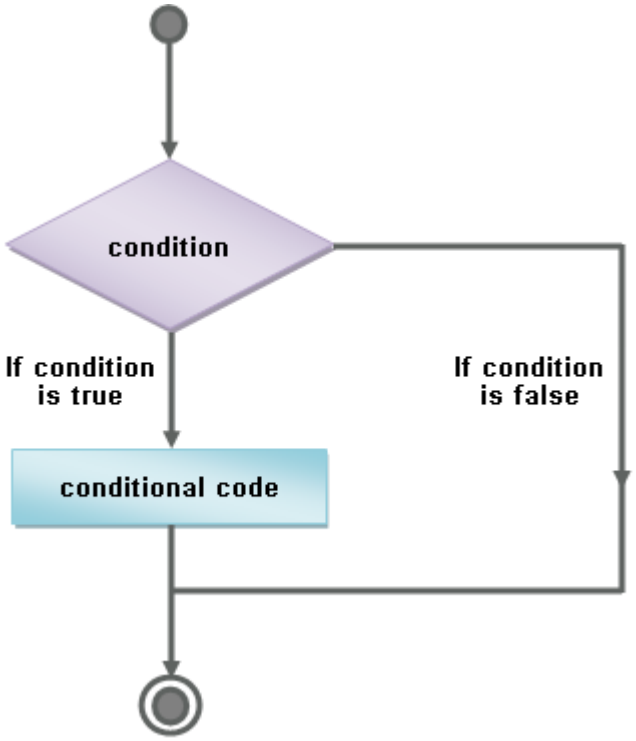
نذكر أن هذه الكلمات المفتاحية لا يمكن استخدامها من أجل تسمية المتحولات.

أولاً، سنعرض شرحاً للعبارات الشرطية Conditional Statements والتي تساعد على اتخاذ قرار decision.

العبارات الشرطية Conditional Statements

تتطلب بنى عملية اتخاذ قرار من المبرمج أن يقوم بتحديد شرط أو مجموعة من الشروط ليتم تنفيذها أو اختبارها ضمن البرنامج، وبناءً على تحقيق الشرط، أو عدم تحقيقه، يتم تنفيذ مجموعة من التعليمات والعبارات، كما يمكن تنفيذ مجموعة من التعليمات في حال تحقق الشرط وتنفيذ مجموعة مختلفة من التعليمات في حال عدم تحقق الشرط.

يعرض الشكل التالي الشكل العام لبنية اتخاذ القرار والموجودة في معظم لغات البرمجة:



يمتلك MATLAB عبارتين أساسيتين تسمحان بالاختيار واتخاذ القرار وهما:

- عبارة if.
- عبارة switch.

عبارة if if statement

تأخذ هذه العبارة عدة أشكال سنقوم استعراضها فيما يلي إضافةً لشرح فائدة كل شكل من الأشكال التالية:

1. if … end

الشكل القواعدي (Syntax) هو:

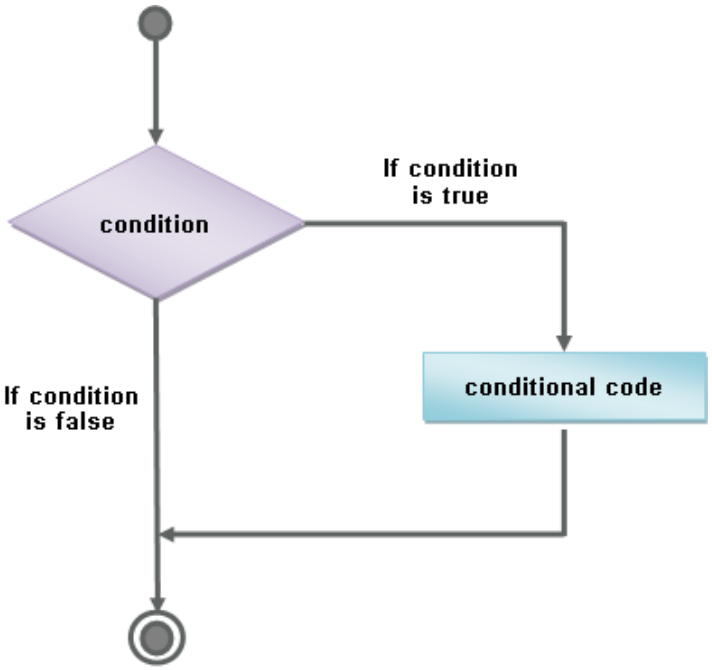
```
if expression
   statements
end
```

if <expression>

```
% statement (s) will execute if the Boolean expression is true
<statements>
end
```

الوصف:

تتألف هذه العبارة من تعليمة if يليها تعبير منطقي يليه مجموعة من التعليمات أو العبارات، في حال تحقق الشرط الموجود ضمن التعبير يتم تنفيذ الرماز البرمجي أما في حال عدم تحقق الشرط لا يتم تنفيذ أي تعليمة ويتم انتقال إلى التعليمات اللاحقة بعد end. يبين الشكل التالي مخططاً للخطوات المنطقية لبرنامج ما يحتوي على العبارة السابقة.



ملاحظة: يجب اتباع الشكل القواعدي السابق أي أن أي تعليمة if يلزمها تعليمة end، في حال نسيانها سينتج خطأ. ملاحظة هامة جداً جداً: يمكن ضمن التعبير expression الموجود ضمن تعليمة if استخدام المؤثرات المنطقية والعلاقاتية Relational and Logical Operators.

ملاحظة: من الآن وصاعداً عند تنفيذ أي مثال يجب إنشاء ملف script وحفظ التعليمات ضمنه لاستخدامها لاحقاً، وفي حال كانت التعليمات عبارة عن تايع يجب إنشاء ملف خاص بالتابع واتباع القواعد التي جرى تعلمها في الفصل السابق.

مثال:

قبل البدء بالمثال ينصح بإنشاء ملف script وتسميته if_end مثلاً، وعند عرض اكثر من مثال يفضل استخدام cells.

يقوم هذا الرماز البرمجي بفحص قيمة متحول إزا كانت أصغر من قيمة ما (عتبة) يظهر رسالة تخبر المستخدم أن قيمة المتحول أصغر من قيمة العتبة ثم تظهر رسالة بقيمة المتحول، في حال كانت قيمة المتحول أكبر من العتبة فإن الرماز يظهر قيمة المتحول فقط.

Editor - C:\Users\Simon\Documents\MATLAB\if_end.m

if_end.m

```
1 -     a = 10;
2       % check the condition using if statement
3 -     if a < 20
4       % if condition is true then print the following
5 -     fprintf('a is less than 20\n' );
6 -     end
7 -     fprintf('value of a is : %d\n', a);
```

عند تنفيذ التابع نجد ما يلي

Command Window

```
a is less than 20
value of a is : 10
fx >>
```

ملاحظة: قم بتغيير قيمة المتحول واجعلها a= 30، ثم قم بتنفيذ الرماز.

مثال 2:

لاحظ تقسيم الرماز إلى خلايا وإضافة التعليقات.

Editor - C:\Users\Simon\Documents\MATLAB\if_end.m

if_end.m | if_else_e...

```
 1       %% if ... end statement
 2       % First Example
 3 -     a = 10;
 4       % check the condition using if statement
 5 -     if a < 20
 6       % if condition is true then print the following
 7 -     fprintf('a is less than 20\n' );
 8 -     end
 9 -     fprintf('value of a is : %d\n', a);
10       %%
11       % Second Example
12 -     question = input (' Do you like MATLAB course? (y/n): ', 's');
13 -     if strcmp (question, 'y')
14 -        disp ('Good Student')
15 -     end
```

عند تنفيذ البرنامج السابق (تذكر يمكن عند الضغط على الخلية cell استخدام ctrl+enter) يظهر مايلي:

Command Window

```
fx Do you like MATLAB course? (y/n):
```

في حالة الإجابة بأي حرف مختلف عن y ضمن command window لا يظهر أي خرج، أما في حالة كتابة y يظهر الخرج التالي:

ملاحظة: سنقدم في فقرات لاحقة من الفصل شرحاً عن التعليمات: fprintf، disp، input.

2. if … else … end

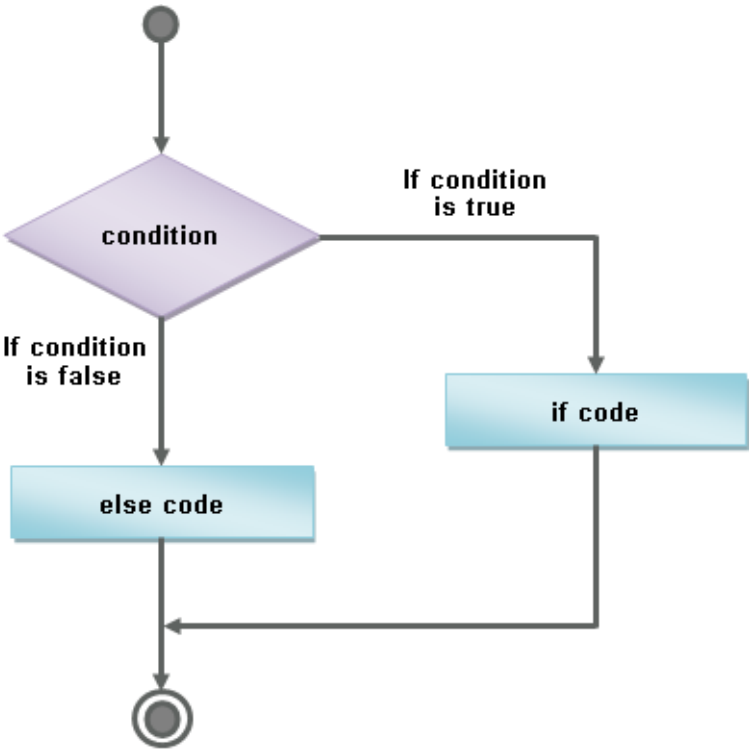
الشكل القواعدي (Syntax) هو:

```
if expression
   statements
else
   statements
end
```

```
if  <expression>
% statement (s) will execute if the Boolean expression is true
<statements>
else
% statement (s) will execute if the Boolean expression is false
<statements>
end
```

الوصف:

تتألف هذه العبارة من تعليمة if يليها تعبير منطقي يليه مجموعة من التعليمات أو العبارات، في حال تحقق الشرط الموجود ضمن التعبير يتم تنفيذ الرماز البرمجي أما في حال عدم تحقق الشرط لا يتم تنفيذ مجموعة التعليمات التي تلي تعليمة else. يبين الشكل التالي مخططاً للخطوات المنطقية لبرنامج ما يحتوي على العبارة السابقة.



مثال:

قم بإنشاء ملف script وتسميته if_else_end مثلاً:

Editor - C:\Users\Simon\Documents\MATLAB\if_else_end.m

```
%% if ... else ... end statement
% First Example
a = 100;
% check the boolean condition
if a < 20
   % if condition is true then print the following
   fprintf('a is less than 20\n' );
else
   % if condition is false then print the following
   fprintf('a is not less than 20\n' );
end
fprintf('value of a is : %d\n', a);
```

عند التنفيذ يكون الخرج:

Command Window

```
a is not less than 20
value of a is : 100
>>
```

قم بتغيير قيمة a إلى a=12، ثم قم بتنفيذ الرماز ستلاحظ الخرج التالي:

```
Command Window
  a is less than 20
  value of a is : 12
fx >>
```

ملاحظة: بعد كتابة الرماز يمكن تحديده كاملاً، والضغط بالزر اليميني لتظهر قائمة من الخيارت ثم يمكن اختيار Smart Indent من أجل تنظيم الرماز بشكل واضح وجعل قرائته وفهمه أسهل.

مثال 2:

```
Editor - C:\Users\Simon\Documents\MATLAB\if_else_end.m
if_else_e... ×
13      %%%
14      % Second Example
15      % Generate a random number
16 -    a = randi(100, 1);
17      % If it is even, divide by 2 and show the result
18 -    if rem(a, 2) == 0
19 -       disp('a is even')
20 -       b = a/2;
21 -       disp('b is equal to')
22 -       disp(b)
23 -    else
24         % else show that a is odd
25 -       disp('a is odd')
26 -    end
```

إن الرماز السابق يولد عدد عشوائي ثم يقوم بفحصه إزا كان زوجياً يظهر رسالة أن العدد زوجي، ثم يقوم بتقسيمه على 2 ويظهر نتيجة القسمة، أما في حال كان فردياً فيظهر رسالة لذلك عند التنفيذ يكون الخرج على شكلين هما:

```
Command Window
  a is even
  b is equal to
     49
fx >>
```

```
Command Window
  a is odd
fx >>
```

3. if … elseif … else … end

الشكل القواعدي (Syntax) هو:

```
if expression
   statements
elseif expression
   statements
else
   statements
end
```

```
if  <expression 1>
% statement (s) will execute if the Boolean expression 1 is true
<statements>
elseif  <expression 2>
% statement (s) will execute if the Boolean expression 2 is true
<statements>
elseif  <expression 3>
% statement (s) will execute if the Boolean expression 3 is true
<statements>
else
% statement (s) will execute when none of the above condition is true
<statements>
end
```

الوصف:

الشكل السابق هو الشكل العام لتعليمة if، حيث وجود تعليمة else وحيدة أو تعليمة elseif وحيدة (أو أكثر) هو اختياري.

إن التعليمات السابقة مفيدة كثيراً من أجل اختبار مجموعة من الشروط على قيم متحول ما لدخل البرنامج او خرج مرحلة ما من الرماز البرمجي.

ملاحظات حول استخدام التعليمات السابقة:

- أي تعليمة if ممكن أن يتبعها تعليمة else وحيدة أو ألا يتبعها أي تعليمة else، وفي حال ورودها يجب أن تأتي بعد تعليمة elseif في حال وجود elseif.
- أي تعليمة if ممكن أن يتبعها تعليمة elseif وحيدة أو اكثر أو ألا يتبعها أي تعليمة elseif، وفي حال ورودها يجب أن تأتي قبل تعليمة else.
- في حال تحقق أي من التعابير ضمن أي واحدة من elseif، فإن أي تعليمة elseif أو else (إضافةص للتعليمات المرافقة لها) لن يتم اختبارهم او تنفيذهم.

مثال:

قم بإنشاء ملف script باسم if_elseif_else_end مثلاً واكتب الرماز التالي:

Editor - C:\Users\Simon\Documents\MATLAB\if_elseif_else_end.m

if_elseif...

```
1       %% First Example
2       % if statements can include alternate choices,
3       % using the optional keywords elseif or else
4
5       % Generate a random integer number between [1- 100]
6 -     a = randi(100, 1);
7 -     if a < 30
8 -       disp('small')
9 -     elseif a < 80
10 -      disp('medium')
11 -    else
12 -      disp('large')
13 -    end
```

الرماز السابق يقوم بتوليد عدد طبيعي عشوائي ضمن المجال [1-100] ثم يقوم بعدة اختبارت وهي:

1. إذا كان العدد أصغر تماماً من 30 تظهر رسالة بكلمة small
2. إذا كان العدد أصغر تماماً من 80 وأكبر أو يساوي 30 تظهر رسالة بكلمة medium
3. إذا كان العدد أكبر أو يساوي من 80 تظهر رسالة بكلمة large

الخرج على الشكل التالي:

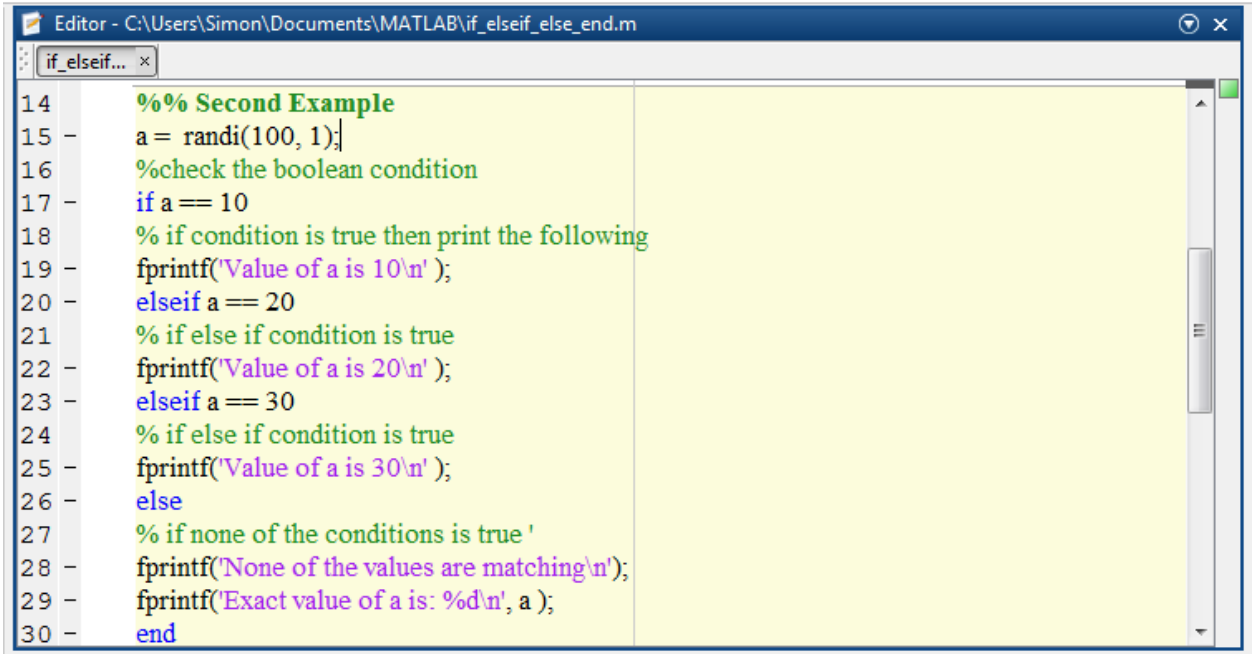
Command Window

```
large
large
small
large
medium
small
medium
medium
large
fx >>
```

يمكن باستخدام workspace مراقبة قيم المتحول، و يمكن طباعة قيمة a في كل مرة باستخدام التعليمة fprintf، قم بالتعرف على التعليمة باستخدام help واستعملها لإظهار قيمة المتحول a.

مثال:

قم بقراءة المثال التالي، وحاول فهم الرماز البرمجي ثم قم بتنفيذه، وحاول ان تفسر نتيجة الخرج.



4. عبارات if متداخلة nested if

تتيح لغة MATLAB خيارات برمجية واسعة للمبرمج، حيث من الممكن كتابة عبارات if ... else متداخلة (nested) أي يمكن استخدام عبارة if (أو elseif) ضمن عبارة if (أو elseif) أخرى.

الشكل القواعدي (Syntax) هو:

```
if <expression 1>
   % Execute when the Boolean expression 1 is true
      if <expression 2>
   % Execute when the Boolean expression 2 is true
      end
end
```

وهو نفس الشكل القواعدي لاستخدام elseif ...else متداخلة.

مثال: قم بإنشاء ملف script باسم nested_if مثلاً واكتب الرماز التالي، وحاول فهم التعليمات ضمنه:

```
Editor - C:\Users\Simon\Documents\MATLAB\nested_if.m*
nested_i... ×
 1 -    a = 100;
 2 -    b = 200;
 3      % check the boolean condition
 4 -    if( a == 100 )
 5        % if condition is true then check the following
 6 -      if( b == 200 )
 7          % if condition is true then print the following
 8 -        fprintf('Value of a is 100 and b is 200\n' );
 9 -      end
10 -    end
11 -    fprintf('Exact value of a is : %d\n', a );
12 -    fprintf('Exact value of b is : %d\n', b );
```

الخرج هو

```
Command Window
   Value of a is 100 and b is 200
   Exact value of a is : 100
   Exact value of b is : 200
fx >>
```

عبارة switch switch statement

الشكل القواعدي (Syntax) هو:

```
switch switch_expression
   case case_expression
      statements
   case case_expression
      statements
    :
   otherwise
      statements
end
```

الوصف:

تقوم كتلة تعليمات switch block بتنفيذ إحدى المجموعات من التعليمات والعبارات بشكل شرطي، من ضمن عدة خيارات choices من مجموعات من التعليمات والعبارات، كل خيار يدعى حالة case.

يأخذ switch_expression ضمن الشكل القواعدي السابق قيم سلمية scalar أو محارف string، أما case_expression فيأخذ قيم سلمية scalar أو محارف string أو خلية صفيفة cell array من العناصر السلمية أو المحارف، عند تنفيذ switch block يتم المرور على كل حالة case حتى يتم تحقق إحداهن وعندها يتم تنفيذ مجموعة التعليمات ضمن الحالة case، ومن ثم يتم الخروج من switch block، إن وضع otherwise هو اختياري ويتم تنفيذ التعليمات التي تليها في حال لم تكن اي من الحالات cases صحيحةً.

ملاحظة: تكون الحالة case صحيحية true في حال كان

- للأعداد السليمة: eq(case_expression, switch_expression) أي متساويين.
- للمحارف: strcmp(case_expression, switch_expression) أي متساويين.

ملاحظة: eq و strcmp تعليمات للمقارنة بين الأعداد، المحارف على الترتيب.

مثال:

قم بإنشاء ملف script وسمّه switch_file مثلاً، واكتب المثال التالي:

Editor - C:\Users\Simon\Documents\MATLAB\switch_file.m

switch_f...

```
1      %% Examples
2      % Example 1
3 -    grade = 'B';
4 -    switch(grade)
5 -       case 'A'
6 -          fprintf('Excellent!\n' );
7 -       case 'B'
8 -          fprintf('Well done\n' );
9 -       case 'C'
10 -         fprintf('Not bad\n' );
11 -      case 'D'
12 -         fprintf('You passed\n' );
13 -      case 'F'
14 -         fprintf('Better try again\n' );
15 -      otherwise
16 -         fprintf('Invalid grade\n' );
17 -   end
```

الخرج هو:

Command Window

```
Well done
>>
```

ملاحظة: قم بتغيير قيمة grade، ودعه يأخذ القيم التالية على الترتيب A,C,Q,D,R,F,Z ولاحظ الخرج لفهم الرماز بشكل أكبر.

مثال:

```
Editor - C:\Users\Simon\Documents\MATLAB\switch_file.m
switch_f... ×
18      %%%
19      % Switch block Conditionally display different text depending
20      % on a value entered at the command line:
21 -    mynumber = input('Enter a number:');
22      % you enter a number as an input
23 -    switch mynumber
24 -       case -1
25 -          disp('negative one');
26 -       case 0
27 -          disp('zero');
28 -       case 1
29 -          disp('positive one');
30 -       otherwise
31 -          disp('other value');
32 -    end
```

الخرج من أجل عدة قيم للدخل:

```
Command Window
Enter a number:5
other value
Enter a number:1
positive one
Enter a number:0
zero
Enter a number:-5
other value
Enter a number:-1
negative one
fx >>
```

ملاحظة: ضمن switch block عند ما يتم تحقق أول حالة case أي أن تصبح صحيحة true يتم تنفيذ التعليمات الخاصة بها دون النظر إلى الحالات التالية حتى لو أن الشرط ضمنها محقق.

مثال:

```
Editor - C:\Users\Simon\Documents\MATLAB\switch_file.m
switch_f... ×
33
34      %%%
35      % In MATLAB switch blocks, only the first matching case executes:
36 -    in = 52;
37 -    switch(in)
38 -       case 52
39 -          disp('result is 52')
40 -       case {52, 78}
41 -          disp('result is 52 or 78')
42 -    end
```

الخرج

```
Command Window
result is 52
fx >>
```

ملاحظة: يمكن استخدام switch متداخلة nested.

ملاحظة: تستخدم عبارة switch بكثرة عندما يريد المستخدم اختبار المساواة لقيمة ما **مع مجموعة محدّدة منتهية ومعروفة من القيم**

مثال: الخرج متضمن مع الصورة

```
Editor - C:\Users\Simon\Documents\MATLAB\switch_file.m
switch_f... ×
42      %%%
43      % when you want to test for equality against a set of known values,
44      % use a switch statement.
45
46 -    [dayNum, dayString] = weekday(date, 'long', 'en_US');
47
48 -    switch dayString
49 -      case 'Monday'
50 -        disp('Start of the work week')
51 -      case 'Tuesday'
52 -        disp('Day 2')
53 -      case 'Wednesday'
54 -        disp('Day 3')
55 -      case 'Thursday'
56 -        disp('Day 4')
57 -      case 'Friday'
58 -        disp('Last day of the work week')
59 -      otherwise
60 -        disp('Weekend!')
61 -    end
Command Window
Last day of the work week
fx >>
```

الحلقات Loops

في معظم الأحيان يحتاج المبرمج إلى تنفيذ مجموعة من التعليمات عدد معين من المرات، يؤمن لنا MATLAB تنفيذ العملية السابقة عن طريق الحلقات Loops، يوجد نوعان من الحلقات ضمن MATLAB:

- الحلقات المعدودة: باستخدام تعليمة for، تقوم هذه التعليمة بتنفيذ سلسلة من التعليمات عدة مرات اعتماداً على دليل الحلقة loop indicator، الذي يحدّد عدد مرات التكرار. يمكن التعبير عن هذه الحلقة بجملة بسيطة:

" كرّر هذه التعليمات عشرين مرة"

- الحلقات الشرطية: باستخدام تعليمة while، تستمر هذه التعليمة بتنفيذ سلسلة من التعليمات طالما أن الشرط المعطى محقق، حيث تقوم باختبار الشرط أولاً ثم تقوم بتنفيذ التعليمات داخل الحلقة. يمكن التعبير عن هذه الحلقة بجملة بسيطة:

" كرّر هذه التعليمات حتى يصبح هذا الشرط خاطئاً"

حلقة while while loop

ذكرنا أن while تستخدم كحلقة شرطية ضمن MATLAB، أي انها تستعمل من أجل تكرار تعليمات محدّدة عندما لا نعرف مسبقاً عدد المرات المطلوب تكرارها فيها.

الشكل القواعدي (Syntax) هو:

```
while expression
   statements
end
```

الوصف:

طالما أن الشرط الموجود ضمن expression لازال محققاً فإن الحلقة تستمر بتنفيذ التعليمات داخل الحلقة، من الضروري عدم نسيان تعليمة end في نهاية حلقة while، وكما هو الحال بالنسبة لتعليمة if فإن التعبير expression من الممكن استخدام المؤثرات المنطقية والعلاقاتية Relational and Logical Operators.

في حالة وضع شرط صحيح دوماً تدخل البرمجية في حلقة لانهائية، من أجل الخروج من هذه الحلقة يمكن استخدام ctrl + c.

مثال: قم بإنشاء ملف script وسمّه while_file مثلاً، واكتب المثال التالي:

في المثال التالي سنقوم بتنفيذ عملية إظهار قيم متحولين طالما أن صحيح، وسنستخدم المؤثرات المنطقية ضمن التعبير.

Editor - C:\Users\Simon\Documents\MATLAB\whileexample.m

whileexample.m

```
1     %%
2 -   a = 5;
3 -   b = 5;
4     % while loop execution
5 -   while ( a < 15 && b < 30)
6 -      fprintf('value of a: %d\n', a);
7 -      fprintf('value of b: %d\n', b);
8 -      a = a + 1;
9 -      b = b + 6;
10 -  end
```

يقوم هذا الرماز بطباعة قيمة المتحولين a و b طالما أن قيمة المتحول a أصغر من 15 و قيمة المتحول b أصغر من 30، **أي أنه يكفي أن يصبح أحد** المتحولات أصغر من القيمة المحدّدة فقط للخروج من الحلقة.

الخرج:

```
Command Window
value of a: 5
value of b: 5
value of a: 6
value of b: 11
value of a: 7
value of b: 17
value of a: 8
value of b: 23
value of a: 9
value of b: 29
fx >>
```

نلاحظ أن المتحول b يصبح أكبر من 30 أسرع من المتحول a وتتوقف الحلقة عن تنفيذ التعليمات.

ملاحظة: جرّب أن يصبح التعبير كالتالي:

(a < 15 || b < 30)

ولاحظ الخرج وفسّر النتيجة.

مثال:

في المثال التالي سنقوم بإيجاد أول عدد صحيح يحقق أن العاملي (Factorial) له أصغر من قيمة معينة، ثم سنقوم بإظهار العدد و أيضاً سنظهر قيمة العاملي الموافقة له.

نذّكر أن العاملي لعدد n هو بالتعريف:

$$n! = n * n-1 * n-2 * \ldots * 1$$

Editor - C:\Users\Simon\Documents\MATLAB\whileexample.m

whileexample.m

```
11      %%
12      % Find the first integer n for which factorial(n) is greater than 520
13 -    n = 1;
14 -    nFactorial = 1;
15 -    while nFactorial < 520
16 -      n = n + 1;
17 -      nFactorial = nFactorial * n;
18 -    end
19 -    fprintf('the first integer n is: %d\n', n);
20 -    fprintf('the factorial(n)  is: %d\n', nFactorial);
```

الخرج:

```
Command Window
the first integer n is: 6
the factorial(n)  is: 720
fx >>
```

ملاحظة: قم بتحويل الرماز السابق إلى تابع يأخذ كدخل له قيمة ولنسميها threshold ويوجد أصغر عدد صحيح يحقق أن قيمة العاملي له أكبر من العتبة المدخلة threshold.

for حلقة for loop

يتم استعمال الحلقة for عندما نحتاج لتكرار تنفيذ مجموعة من التعليمات أو تابع ما **عدد محدّد ومعروف مسبقاً من المرات**، ما يتم ضمن الحلقة هو مفهوم التكرار iterate، حيث يوجد متحول يتم استعماله من أجل المرور على القيم يدعى هذا المتحول كما أشرنا سابقاً بمتحول (دليل، عداد) الحلقة Loop indicator

الشكل القواعدي (Syntax) هو:

```
for index = values
   program statements
        :
end
```

```
for index = values
   program statements
        :
end
```

مثال:

قم بإنشاء ملف script وسمّه for_file مثلاً، واكتب المثال التالي:

مثال 1: يبين لنا حلقة تقوم بعرض مجموعة من القيم بخطوة 1.

```
for a = 2 : 1 : 9
    fprintf('the value of a is: %d\n',a)
end
```

ملاحظة: يمكن كتابة a = 2:9 لأن الخطوة الافتراضية ضمن البرمجية هي 1.

مثال 2: مماثل للمثال السابق إلا أن العداد يعد بخطوة 2. الناتج هو الأعداد الزوجية

```
for a = 0 : 2 : 10
    fprintf('the value of a is: %d\n',a)
end
```

ملاحظة: قم بالتغيير على شعاع القيم والإبقاء على الخطوة 2 وقم بعرض الأعداد الفردية.

مثال 3: مماثل للمثال السابق إلا أن العداد يعد بخطوة سالبة.

```
for d = 2 : -0.2 : 0
    fprintf('the value of a is: %0.2f\n',d)
end
```

مثال 4: نقوم بعرض قيم شعاع.

```
for a = [2,180,317,3,208]
   disp(a)
end
```

ملاحظة هامة: ذكرنا في الفصول السابقة أن عملية indexing للمصفوفات تقبل فقط أعداد موجبة أي أن العنصر الأول يكون الدليل له هو 1 وليس 0، لذلك عند تعريف مجال قيم العداد يجب الانتباه إلى موضوع الفهرسة

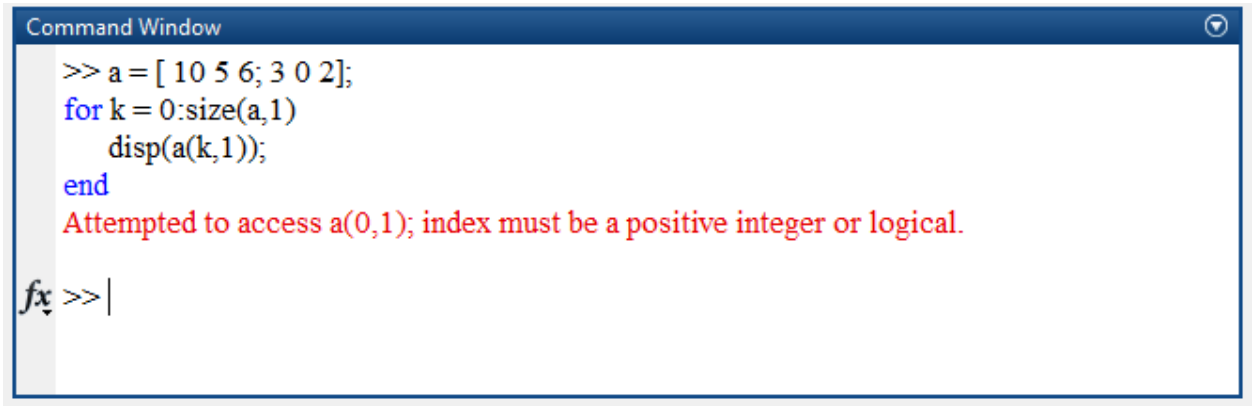
مثال على الملاحظة السابقة:

نرغب في عرض قيم عمود مصفوفة قمنا بتعريفها مسبقاً، سنقوم بعرض قيم العمود الأول مثلاً:

اكتب الرماز التالي:

```
a = [ 10 5 6; 3 0 2];
for k = 0:size(a,1)
    disp(a(k,1));
end
```

قم بتنفيذه ولاحظ ظهور الخطأ التالي:



لتصحيح الرمازالسابق قم بتغيير القيمة البدائية للعداد لتصبح على الشكل:

```
a = [ 10 5 6; 3 0 2];
for k = 1:size(a,1)
    disp(a(k,1));
end
```

ملاحظة: حاول عند استعمال متحولات لعداد الحلقة (متحول التكرار) الابتعاد عن المحارف i و j فهذه المحارف كما رأينا مسبقاً مخصصة لتعريف الأعداد العقدية، وفي حالة استخدامهم كعداد للحلقة لايمكن تعريف العدد العقدي باستخدام الحرف الذي قمت باستخدامه أي إزا استخدمت المحرف i ضمن الحلقة قم بتعريف الأعداد العقدية باستخدام المحرف j.

ملاحظة: تذكر الفرق دوماً بين المؤثرات المخصصة لجداء ،قسمة ، الرفع إلى قوة، ... المخصصة للمصفوفات وتلك المؤثرات المخصصة لجداء ،قسمة ، الرفع إلى قوة، ... الصفيفات والتي تستخدم (.) دوماً أي التي تعمل على مستوى عنصر لعنصر.

ملاحظة: كما رأينا في فقرة if، أنه يمكن استخدام تعليمات if متداخلة، nested if، كذلك الأمر بالنسبة لتعليمات for و while، نعرض هنا الشكل القواعدي (Syntax):

```
for m = 1:j
    for n = 1:k
        <statements>;
    end
end
```

```
while <expression1>
   while <expression2>
       <statements>
   end
end
```

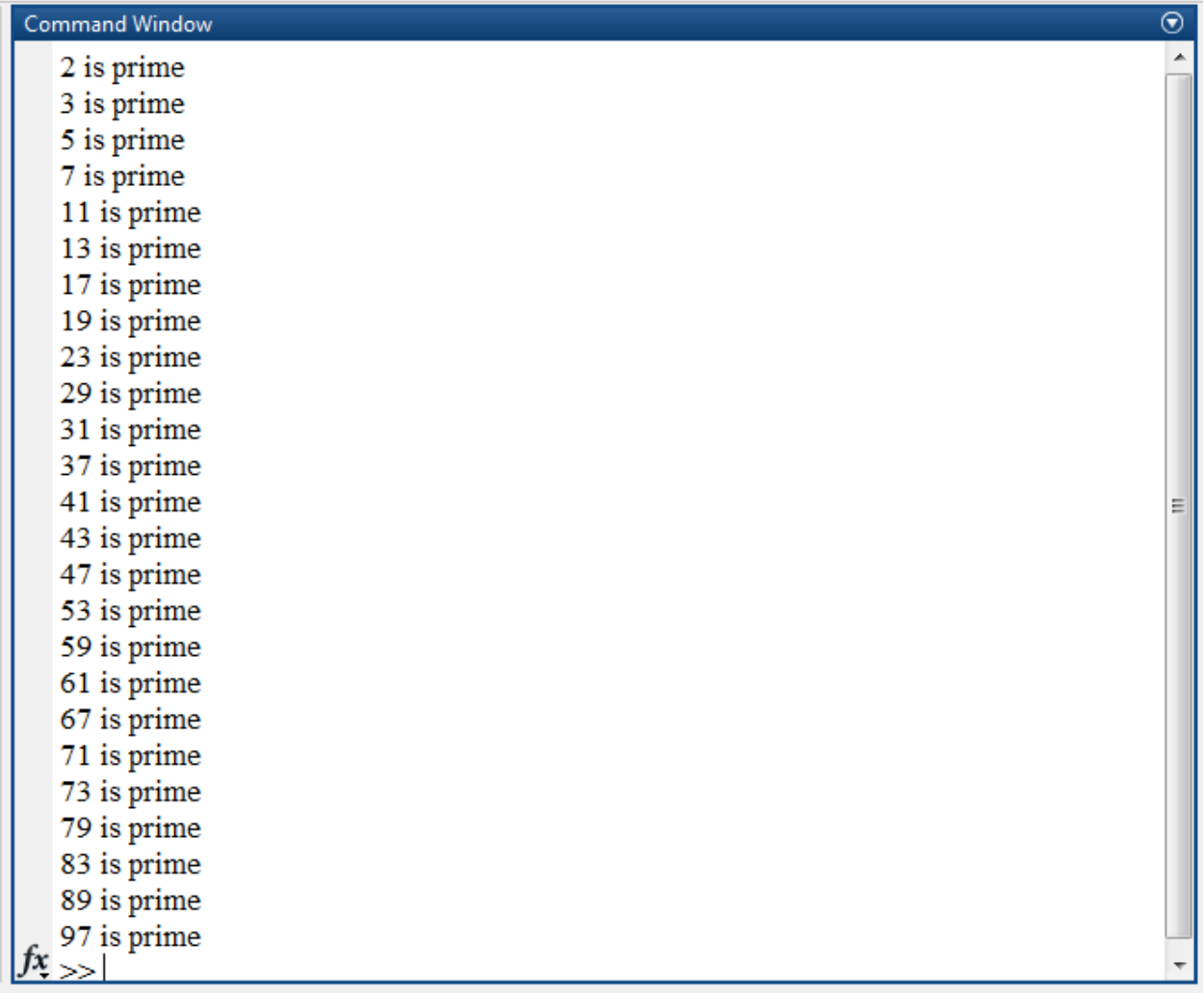
مثال للملاحظة السابقة: سنستخدم في هذا المثال حلقات for متداخلة إضافةً لبعض الشروط لعرض كل الأعداد الأولية ضمن المجال [1-100].

Editor - C:\Users\Simon\Documents\MATLAB\for_file.m

for_file.m

```
15      %%
16 -    for i=2:100
17 -      for j=2:100
18 -        if(~mod(i,j))
19 -          break; % if factor found, not prime
20 -        end
21 -      end
22 -      if(j > (i/j))
23 -        fprintf('%d is prime\n', i);
24 -      end
25 -    end
```

الخرج:



ملاحظة هامة جداً: يرجى الاطلاع على عمل التعليمات break, try/catch, continue.

ملاحظة هامة جداً جداً:

أوضحنا في الفقرة السابقة كيفية استخدام حلقات for، إلا أنه يجب التذكير إلى ان MATLAB مبني أساساً على المصفوفات ويستخدم العمليات الحسابية على المصفوفات أي أن التوابع المبنية ضمن مكاتب البرمجية تتعامل مع المصفوفات والعمليات عليها وقد جرى عمليات أمثلة optimization عليها لتنفيذ المطلوب بزمن أقل، لذلك ينصح بعدم استخدام الحلقات من أجل إجراء عمليات حسابية تتعلق بالمصفوفات، مثلاً جرّب الرماز التالي:

```
%%

clear all;close all;clc;

a = randi(100,1,10000);

b = randi(100,1,10000);

disp([char(10),'Using Loops:'])

tic;

for i=1:10000
```

```
        c(i)= a(i).*b(i);
end
toc
disp([char(10), 'Using Array Operations:'])
tic
c1=a.*b;
toc
```

الخرج هو:

نلاحظ الفرق في زمن التنفيذ.

التعليمات tic ثم toc تستخدم لمعرفة الزمن اللازم لتنفيذ رماز برمجي، اولاً نضع tic في بداية الرماز الذي نرغب بتقدير الزمن تقوم هذه التعليمة بإنشاء عداد زمني واحدته ميلي ثانية millisecond ثم نضع في نهاية الرماز تعليمة toc والتي بدورها تقوم بإعادة الزمن المنقضي time elapsed منذ تفعيل آخر تعليمة tic.

يعرف المفهوم السابق بـvectorization أي التحويل إلى أشعة vector وهو ما ينصح به عند استخدام MATLAB أي تجنب الحلقات قدر الإمكان والتفكير لتحويل المسألة إلى عمليات مصفوفات للاستفادة من سرعة التوابع المنجزة في المكاتب.

مثال:

```
%%
clear all;close all;clc;
% Define Variables
Mass = rand(5,1000);
Length = rand(5,1000);
Width = rand(5,1000);
Height = rand(5,1000);
% get the size
```

```
[rows, cols] = size(Mass);
disp([char(10), 'Using Array Operations:'])
tic
Density = Mass./(Length.*Width.*Height);
toc
disp([char(10),'Using Loops:'])
tic;
for I = 1:rows
  for J = 1:cols
    Density(I) = Mass(I,J)/(Length(I,J)*Width(I,J)*Height(I,J));
  end
end
toc
```

الخرج عند التنفيذ:

ملاحظة: عند استخدام الحلقة من أجل حساب مجموع لمتحول ما (قيمة سلمية، شعاع، مصفوفة)، أي إضافة مجموعة من القيم للمتحول، يجب عدم نسيان إعطاء قيمة بدائية للمتحول وهي 0 في حالة متحول سلمي أو شعاع (أو مصفوفة) من الأصفار (باستخدام تعليمة zeros) في حالة متحول هو شعاع (أو مصفوفة)، تفيد هذه العملية في تقليل الزمن اللازم لتنجير رماز برمجي. مثلاً لنقم بالتعديل على المثال الأول ضمن هذه الملاحظة:

```
%%
clear all;close all;clc;
a = randi(100,1,10000);
b = randi(100,1,10000);
c=zeros(1,10000);
c1=zeros(1,10000);
```

```
disp([char(10),'Using Loops:'])
tic;
for i=1:10000
        c(i)= a(i).*b(i);
end
toc
disp([char(10), 'Using Array Operations:'])
tic
c1=a.*b;
toc
```

الخرج هو

لاحظ نقصان زمن التنفيذ بشكل كبير تعتبر الملاحظتين السابقتين من أهم الملاحظات البرمجية عند استخدام MATLAB يضاف لها ملاحظة بسيطة وهي استخدام (;) لمنع إظهار الخرج الغير لازم ضمن command window، فعملية الإظهار تلك تزيد من زمن التنفيذ ويجب الاكتفاء بإظهار النتائج المهمة والمعبرة فقط، يمكن استخدام توابع disp، fprintf مثلاً.

النصيحة الأخيرة هي تغيير نمط المتحول، عند إجراء بعض الحسابات لا نحتاج دائماً إلى دقة كبيرة في التمثيل لذلك من الممكن عند إجراء عمليتا حسابية عادية أو أننا لا نحتاج إلى دقة كبيرة في النتيجة يمكن استخدام أنماط للمتحولات تتطلب حجماً أصغر في الذاكرة، تؤثر العملية السابقة على زمن الحسابات وعلى الذاكرة المستخدمة (المحجوزة) عند تنفيذ الرماز.

السلاسل المحرفية Strings

تعاملنا في الفصل الثاني قليلاً مع المحارف chars والسلاسل المحرفية strings، ووجدنا أن إنشاء متحولات من نمط string أو char هو أمر بسيط، نهدف في هذه الفقرة غلى عرض بعض الأمور الإضافية المتعلقة بالتعامل مع السلاسل المحرفية، واستخدام التوبع المبنية في MATLAB والتي تسهل عمل المبرمج و المستخدم.

تذكرة سريعة:

لإنشاء متحول نقوم بالتعريف على الشكل التالي:

```
str='Hello MATLAB students !!!';
```

ونذكر بأن التعريف السابق ينشئ متحول عبارة عن شعاع سطر من النمط char ويكون الحجم من الذاكرة اللازم لكل محرف هو 2 Bytes.

من أجل دمج المحارف concatenate استخدم [str1 , str2] مثلاً، اكتب الرماز التالي:

```
str1='Hi there.  '

str2='How are you?'

str3= 'Bye'

c1= [str1,' ',str2]  % join two strings
```

الدمج السابق يدعى horizontal concatenation، أما من أجل دمج محارف بشكل عمودي vertical concatenation هنا يجب التحقق من أن المتحولات لها نفس الطول، جرّب:

```
c2 = [str1;str2]  % vertical concatenation (must be same length)
```

ثم

```
C_err= [str1;str3]
```

ستظهر رسالة خطأ.

يمكن استخدام التابع strvcat من أجل الحصول على vertical concatenation، جرّب:

```
c3=strvcat(str1,str2,str3)  % vertically concatenate (matrix)
```

قم باستخدام help وفق المسار التالي للتعرف أكثر على string وكيفية دمج concatenate أكثر من محرف:

MATLAB → Language Fundamentals → Data Types → Characters and Strings → Create and Concatenate Strings

ملاحظة: يتم التعامل مع السلاسل المحرفية في MATLAB على انها أشعة أي يمكن تطبيق كافة العمليات والتوابع المستعملة مع الأشعة على السلاسل المحرفية مثلاً يمكن الحصول على منقول الشعاع، او استخدام التابع length مثلاً.

يمتلك MATLAB مجموعة من التوابع المفيدة كثيراً عند التعامل مع السلاسل المحرفية والتي تسهل عمل المبرمج وتفيد في عدم ضياع الوقت في إعادة برمجة مثل هذه التوابع الضرورية والأساسية نذكر منها:

Command التعليمة	Description الوصف
strcat	تقوم بدمج السلاسل المحرفية

ischar,isletter, isspace	تنفيذ بالتحقق فيما إزا كان المتحول هو محرف، حرف أبجدي، فراغ space.
int2str, num2str	تقوم هذه التعليمات بالتحويل من متحول نمطه int إلى string أو من number إلى string.
strcmp	للمقارنة بين المحارف.
sprintf,fprintf	العودة إلى استدعاء التابع.

قم باستخدام help للتعرف بشكل أكبر على التعليمات sprintf, fprintf ثم قم بالبحث وفق المسار التالي للتعرف أكثر على فقرة formatting strings والتي تساعد في فهم التعليمات السابقة بشكل أكبر

MATLAB → Language Fundamentals → Data Types → Characters and Strings → Formatting Strings

مثال: اكتب الرماز التالي، ولاحظ الخرج وحاول أن تفسره اعتماداً على القراءة السابقة.

```
C =2* pi*1000*ones(1,6);

sprintf(' %f \n %.2f \n %.4f \n %+.2f \n %12.2f \n %012.2f \n', C)
```

الخرج هو:

ملاحظة هامة: يرجى الانتباه بعد التعلم عن تعليمتي fprintf و sprint أن الفرق الجوهري هو أن التعليمة sprint تنشئ سلسلة محرفية أما التعليمة fprintf فتقوم بطباعة السلاسل المحرفية فقط.

البيانيات والإظهار Graphics and visualization:

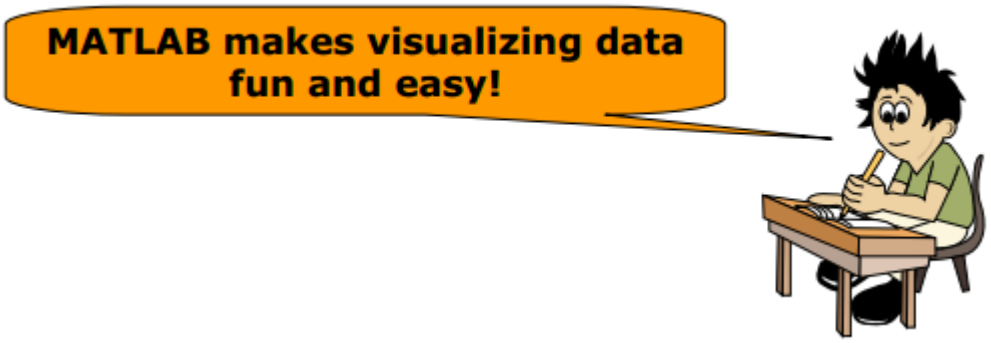
قدمنا في الفصول والفقرات السابقة فقرات عديدة تناقش تعريف متحولات على اختلاف نمطها، نوعها وحجمها، إلا أننا لم نتطرق لكيفية إظهار هذه المتحولات؛ وكما هو معروف أن الرسوم البيانية في كثير من الأحيات تعطي صورة أفضل عن النتيجة، كما ان رسم بعض التوابع او المتحولات يساعد في فهم سلوك المتحول المعبر عن ظاهرة ما.

نعرض في هذه الفقرة عملية الإظهار ضمن MATLAB وكافة المواضيع والتعليمات المتعلقة بالتصوّر visualization حيث سنركز على:

- 2-D Plotting
- Graph Annotation
- Subplots
- Alternative Axes
- 3-D Plotting
- Saving and Exporting Figures

بشكل عام لرسم Plot شكل تابع $y = f(x)$ نحتاج إلى الخطوات التالية:

- تعريف شعاع الدخل للتابع أو ما يعرف بالمنطلق، x، وهنا المقصود هو تحديد مجال القيم للمتحول x التي نرغب في معرفة قيم التابع عندها، ومن ثم رسمه.
- تعريف التابع $y = f(x)$.
- استخدام التوابع المبنية ضمن MATLAB والتي تفيد في رسم منحني التابع، وهي عديدة منها ما هو مستمر ومنها ما هو متقطع، ومنها ما هو مخصص لرسمات ثنائية الأبعاد 2D Plot أو ثلاثية الأبعاد 3D Plot، نذكر منها: plot, stem, stair, plot3.



مثال:

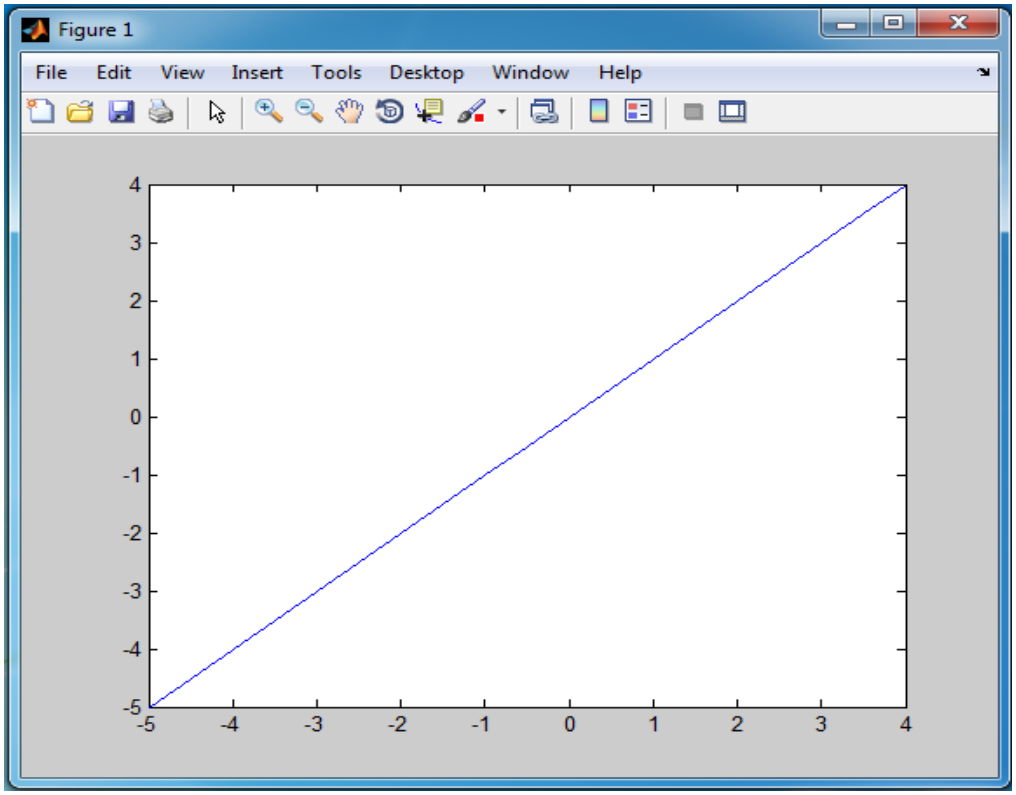
سنقوم في هذا المثال برسم التابع $f(x) = x$ على المجال $[-5, 4]$ بخطوة مقدارها 0.5. قم بإنشاء ملف script واكتب ضمنه الرماز التالي:

```
x = -5 : 0.5 : 4;

y=x;
```

```
plot(x,y)
```

ستلاحظ إنشاء شكل يعبر عن الخرج وهو:

ملاحظة: يتيح لنا MATLAB القيام بخيارات إضافية عند ظهور شكل figure، بعد تنفيذ مجموعة من التعليمات، منها طباعة الشكل، نسخ وحفظ الشكل وذلك بلواحق عديدة سواء كملفات أشكال MATLAB بلاحقة fig. أو كصور بلاحقة png. tif. jpg. bmp.، إضافة ملاحظات للشكل الناتج مثل إشافة تسمية للمحاور، عنوان أو ملاحظات نصية (دون استخدام التعليمات) ويمكن أيضاً القيام بتدوير للشكل، zoom in، zoom out. نجد هذه الإضافة ضمن الشرط بأعلى الشكل قم بالضغط على File أو Insert أو Tools للتعرف على الخيارات بشكل أكبر.



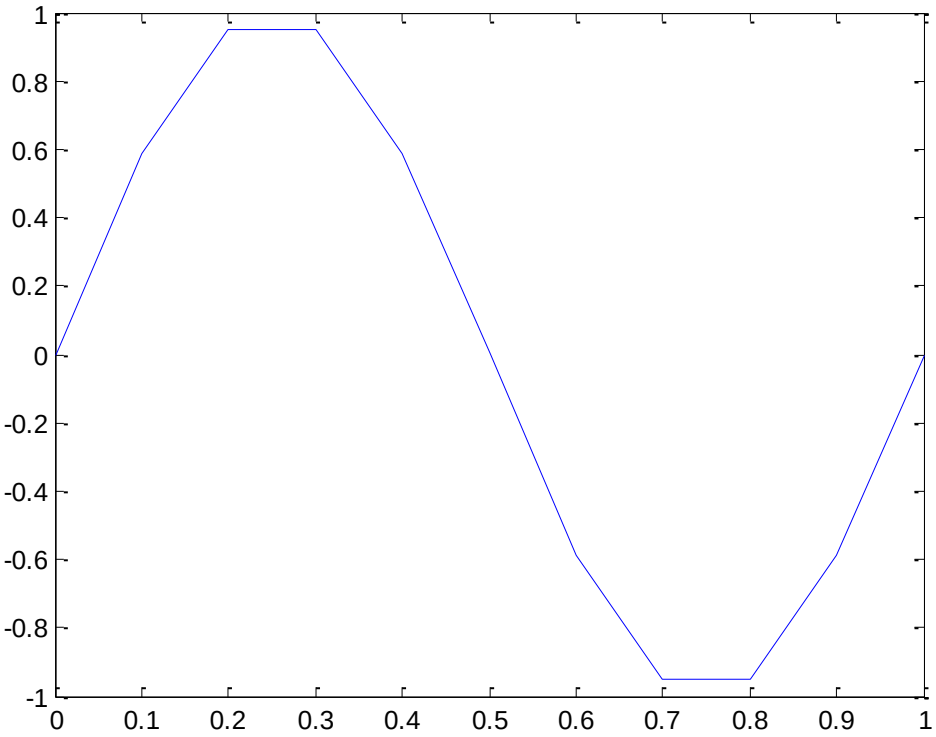
سنأخذ المثال التالي، الهدف منه هو توضيح أهمية أخذ عدد نقاط كافي من أجل الحصول على الأشكال بشكل صحيح، حيث أن برمجية MATLAB تقوم بعملية الاستيفاء Interpolation بين النقاط باستخدام توابع خطية هي خطوط مستقيمة ومنه كلما ازداد عدد النقاط اقترب الشكل الناتج من الواقع.

لنأخذ المثال التالي، حيث نرغب في رسم تابع Sin(x). نكتب الرماز الأول

```
%%
clear all;
close all;
```

```
clc;
x_in= 0:0.1:1;
y=sin(2*pi*x_in);
plot(x_in,y)
```

سيظهر الشكل التالي:

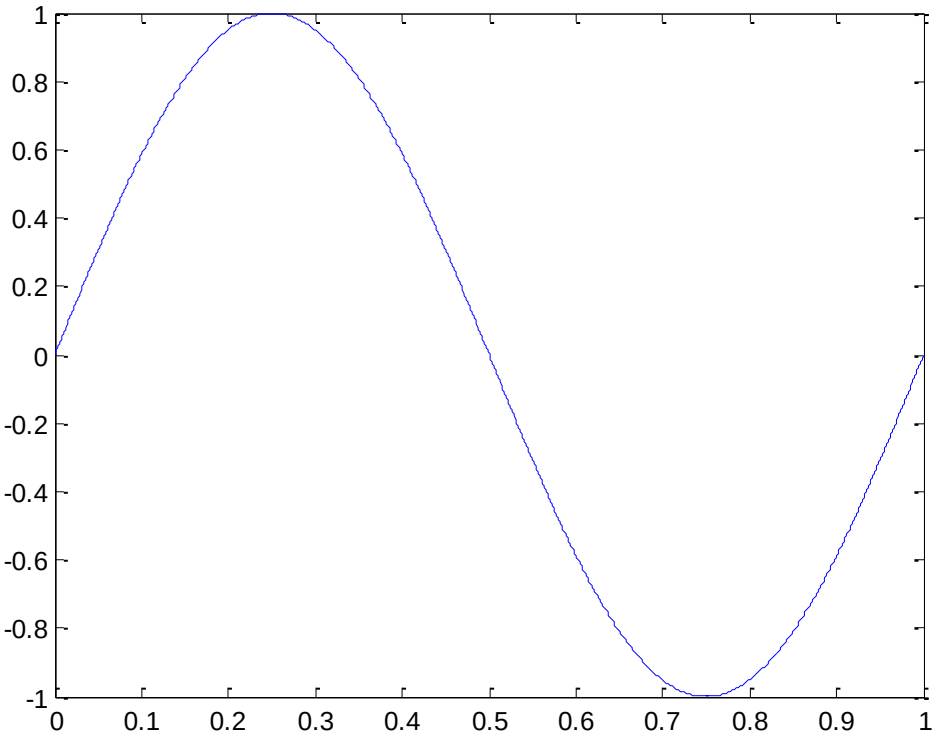


نلاحظة كيفية أن التابع السابق قريب من التابع sin(x) لكنه ليس صحيحاً مثلاً لاحظ ثبات القيمة بين النقطتين x=0.2 و x=0.3 كما أن قيمة التابع نظرياً محدودة ضمن المجال [-1,1] في حين هنا أعلى قيمة يصلها لها التابع هي $x \cong 0.95$ والتفسير هو قلة عدد النقاط المستخدمة لرسم التابع (11 نقطة).

نغير قيمة شعاع الدخل x_in في المثال السابق ليصبح عدد النقاط 1001 نقطة على الشكل التالي:

```
x_in= 0:0.001:1;
```

ثم نعيد تنفيذ الرماز السابق فنحصل على الشكل التالي:



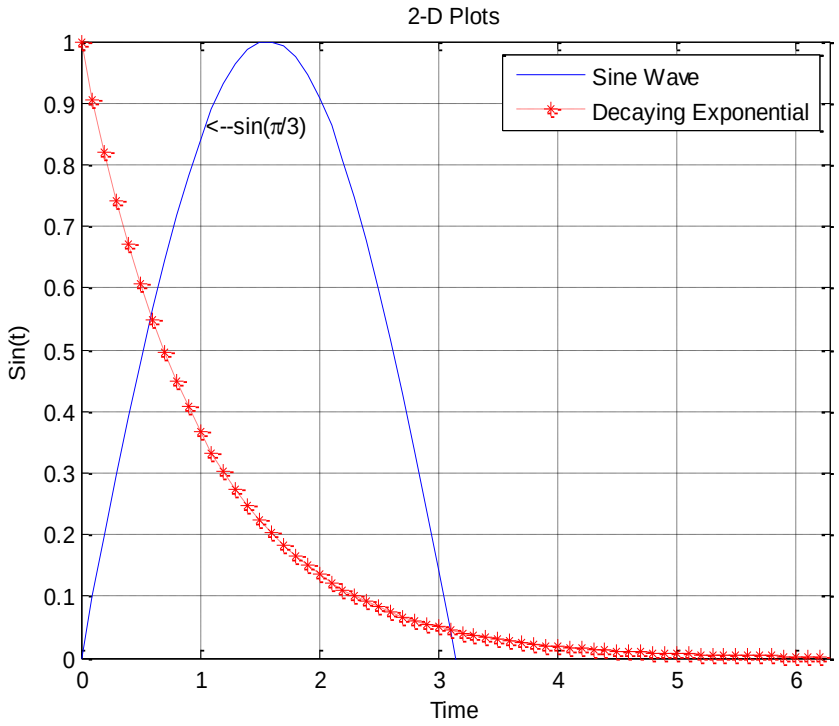
مثال 2D Plot and Graph Annotation :

باستخدام ملف script السابق قم بإنشاء خلية جديدة، ثم اكتب الرماز التالي، وحاول فهم الرماز حيث تمت إضافة تعليقات لتسهيل فهم الرماز:

```
%%
x = 0:0.1:2*pi; % Define input desired variable range
y=sin(x); % Define the function sin(x)
plot(x,y); %Make 2D plot
grid on; % Intialize grid lines to the current axes
hold on; % Allow user to plot another plot on the same figure
plot(x , exp(-x),'r:*'); %Make 2D plot
axis ([ 0 2*pi 0 1]) % Define scope of view range: axis([xmin xmax ymin ymax])
title('2-D Plots');% Adds a title to the plot
xlabel('Time');% Adds an x axis label
ylabel('Sin(t)');% Adds an y axis label
text(pi/3,sin(pi/3),'<--sin(\pi/3)') %Add text to a specified position on the plot
```

```
legend('Sine Wave','Decaying Exponential');% Add a legend  to the plot
```

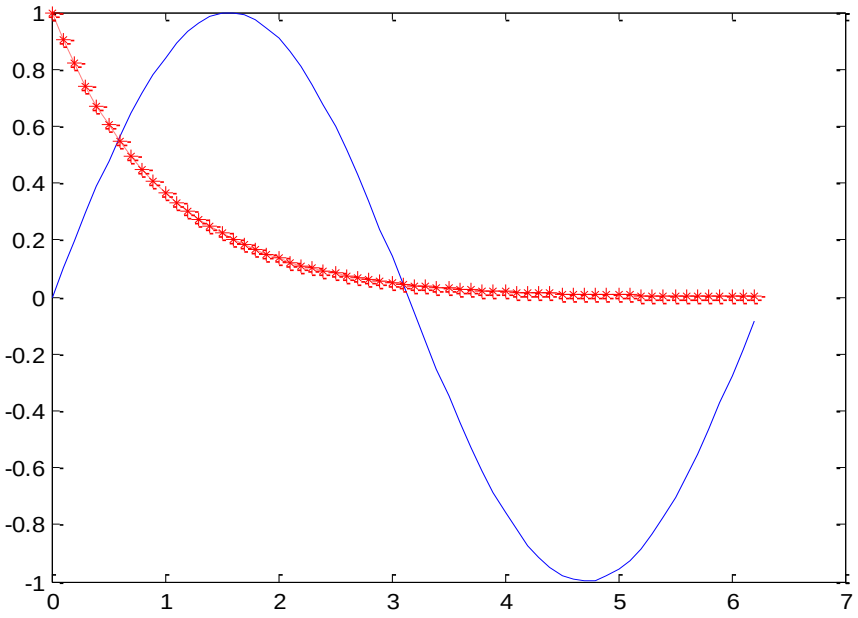
الخرج على الشكل التالي:



ثم قم بكتابة الرماز التالي:

```
figure(2);
plot(x,y,x,exp(-x),'r:*')
```

لتحصل على الشكل التالي:

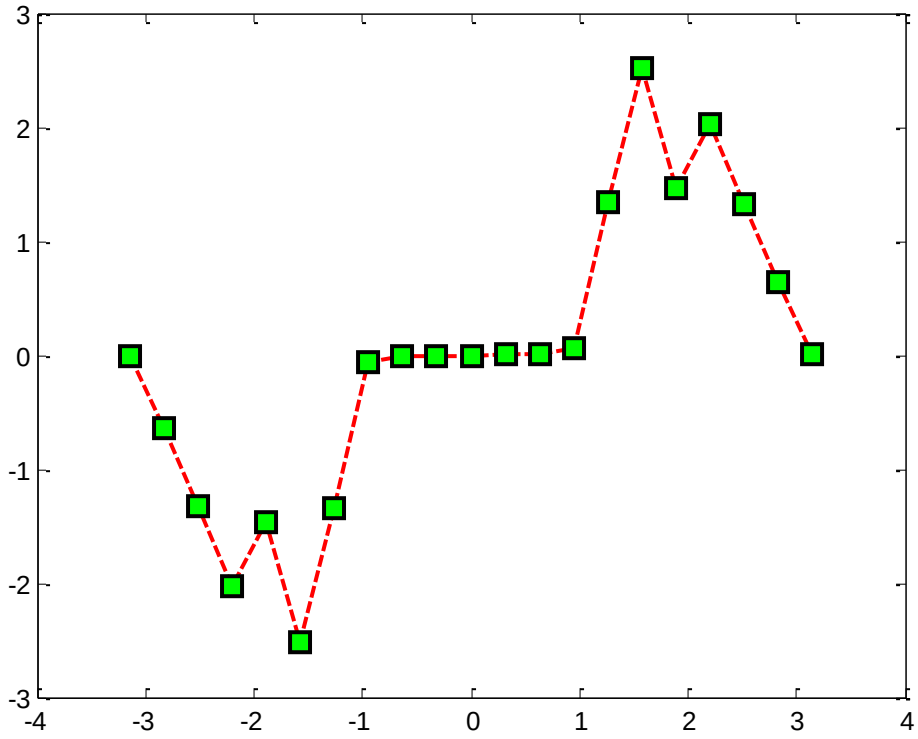


تقوم تعليمة figure(); بإنشاء شكل جديد لرسم المنحني ضمنه، لاحظ ان تعليمة plot تسمح برسم أكثر من تابع أي تسمح بإدخال أكثر من ثنائية من (x,y) , (x1,y1)... (xn,yn).

لنتعرف على خيارات أكثر ضمن تابع plot مثلاً لاحظ الشكل الناتج بعد إضافة الخيارات ضمن التعليمة plot

```
%%
x = -pi:pi/10:pi;
y = tan(sin(x)) - sin(tan(x));
plot(x,y,'--rs','LineWidth',2,...
              'MarkerEdgeColor','k',...
              'MarkerFaceColor','g',...
              'MarkerSize',10)
```

الخرج:



يمكن تغيير الألوان للأشكال الناتجة وذلك بهدف التمييز بينها وذلك بإضافة الرماز الخاص بكل لون ضمن تعليمة Plot على الشكل التالي:

```
plot(x,y,'r')
```

يعرض الجدول التالي الألون والرمازات الموافقة لها

Color	Code
White	w
Black	k
Blue	b
Red	r
Cyan	c
Green	g
Magenta	m
Yellow	y

يجب استخدام help للتعرف أكثر على التعليمات المستخدمة في الأمثلة السابقة

مثال subplot:

سنستخدم في هذا المثال تعليمة subplot والتي تتيح للمستخدم بعرض عدة رسوم على نفس الشكل حيث يتم تقسيم الواجهة الخاصة بعرض الأشكال figure. التعليمة تأخذ المعاملات التالية subplot(m,n,p) :

يتم تقسيم الشكل الناتج إلى مصفوفة من النوافذ أبعادها $m \times n$ ويستخدم المتحول p لإعطاء موضع الشكل المراد إظهاره ضمن النافذة الكلية أي هو دليل الموقع ضمن مصفوفة التقسيم.

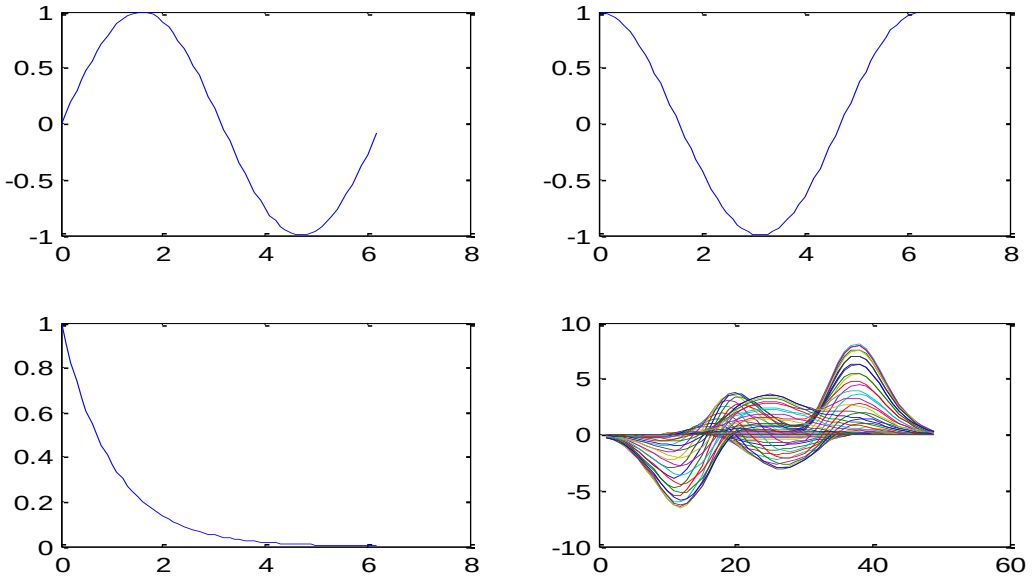
نذكر أن تعليمة subplot تقوم فقط بتجزئة النافذة ولا تقوم بالرسم حيث يجب استخدام تعليمات الرسم ثنائية الأبعاد مثلاً مثل plot.

لنكتب الرماز التالي بعد إنشاء خلية جديدة ضمن نفس ملف script، استخدم التعليمة clear all لمسح جميع المتحولات المعرّفة مسبقاً، والتعليمة clc لمسح نافذة Command window، اما close all فتستخدم لإغلاق جميع نوافذ الأشكال التي جرى تنفيذها مسبقاً، التعليمات الثلاثة السابقين ينصح باستخدامهم بشكل دائم عند كتابة رماز برمجي وذلك لضمان عدم استخدام متحولات جرى تعريفها مسبقاً ولمعرفة جميع الأشكال الناتجة عن تنفيذ الرماز المحدّد ومسح التعليمات من نافذة command window.

```
%%
% subplot - display multiple plots in the same window
% subplot (nrows,ncols,plot_number)
clear all;close all;clc;
x=0:.1:2*pi;
subplot(2,2,1);
plot(x,sin(x));
subplot(2,2,2);
```

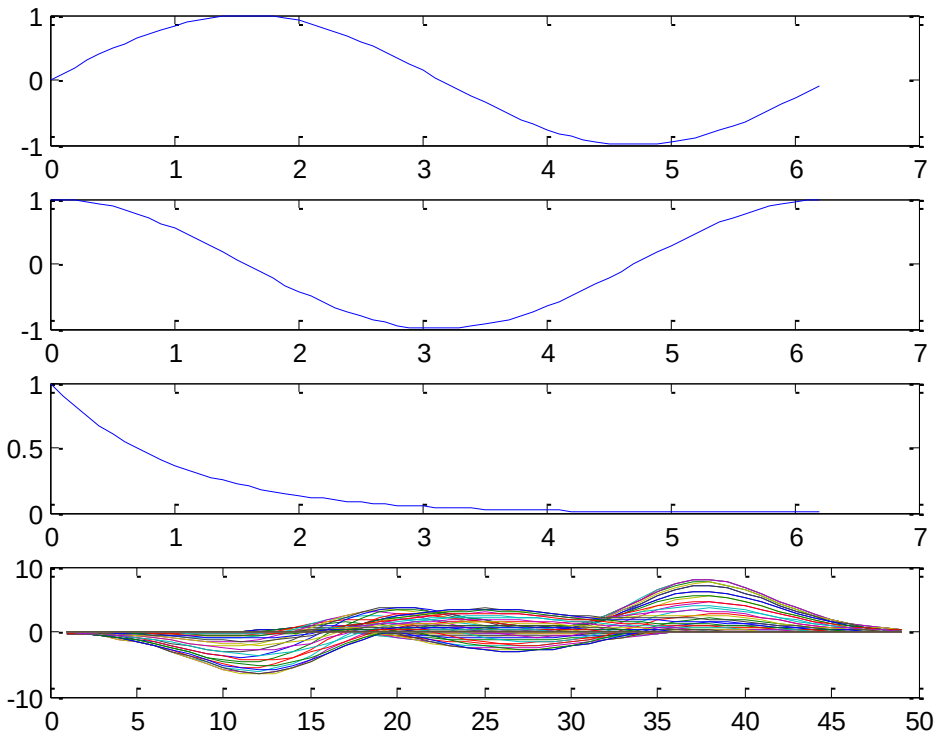
```
plot(x,cos(x));
subplot(2,2,3)
plot(x,exp(-x));
subplot(2,2,4);
plot(peaks);
```

الخرج على الشكل:



لفهم التعليمة بشكل أكبر قم بكتابة الرماز التالي، ولاحظ الفروقات مع الرماز السابق وانظر إلى الخرج.

```
subplot(4,1,1);
plot(x,sin(x));
subplot(4,1,2);
plot(x,cos(x));
subplot(4,1,3)
plot(x,exp(-x));
subplot(4,1,4);
plot(peaks);
```



مثال scale for axes:

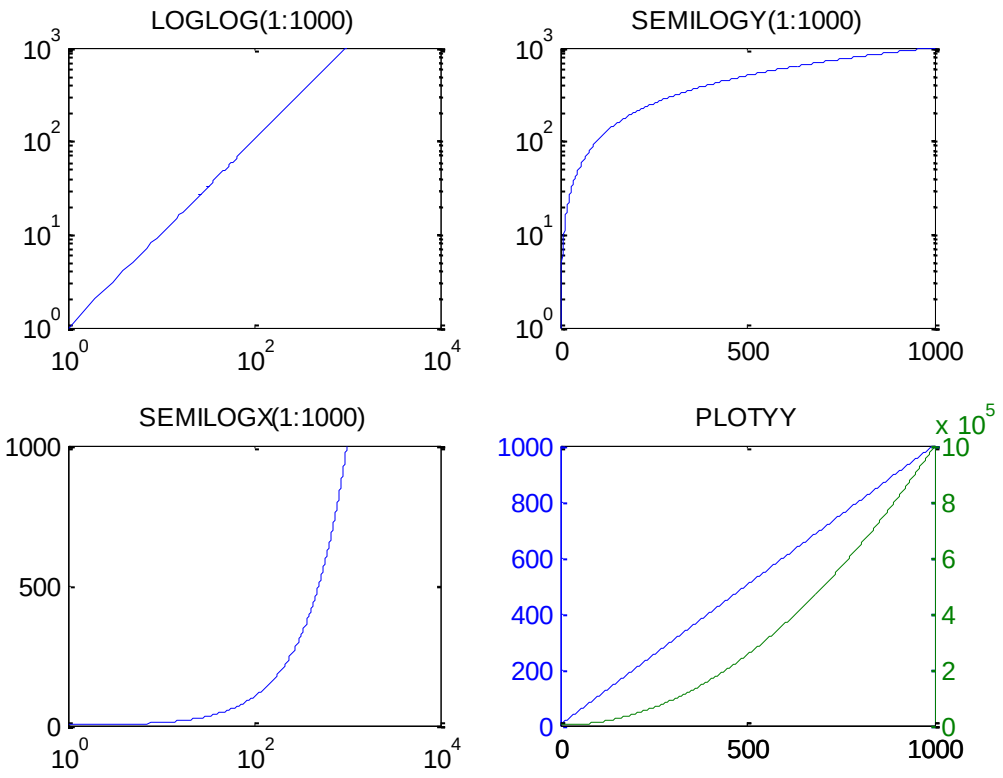
نهتم في هذا المثال بعرض كيفية تغيير التقييس لمحاور الرسم، فمثلاً يوجد رسم على محاور سلمية، او لوغاريتمية أو نصف لوغاريتمية بالنسبة لمحور شعاع الدخل او محور شعاع الخرج، يوضح المثال التالي هذه الانواع:

نكتب الرماز التالي بعد إنشاء خلية جديدة ضمن نفس ملف script:

```
clear all;
close all;
clc;
data = 1:1000;
subplot(2,2,1)
loglog(data)
title('LOGLOG(1:1000)')
subplot(2,2,2)
semilogy(data)
title('SEMILOGY(1:1000)');
subplot(2,2,3)
```

```
semilogx(data)
title('SEMILOGX(1:1000)')
subplot(2,2,4)
plotyy(data,data,data,data.^2)
title('PLOTYY');
```

الخرج على الشكل:



تستخدم تعليمة plot من أجل رسم منحنيات باستخدام محاور تم تقييسها بأساس خطي linear.

تستخدم تعليمة loglog من أجل رسم منحنيات باستخدام محاور تم تقييسها بأساس لوغاريتمي log (أساسه 10).

تستخدم تعليمة semilogx من أجل رسم منحنيات حيث المحور x تم تقييسه بأساس لوغاريتمي log10 أما المحور y تم تقييسه بأساس خطي.

تستخدم تعليمة semilogy من أجل رسم منحنيات حيث المحور y تم تقييسه بأساس لوغاريتمي log10 أما المحور x تم تقييسه بأساس خطي.

تستخدم تعليمة plotyy من أجل رسم مجموعتين من المعطيات باستخدام نفس المحور x لكن من أجل مقياسين مختلفين على المحور y.

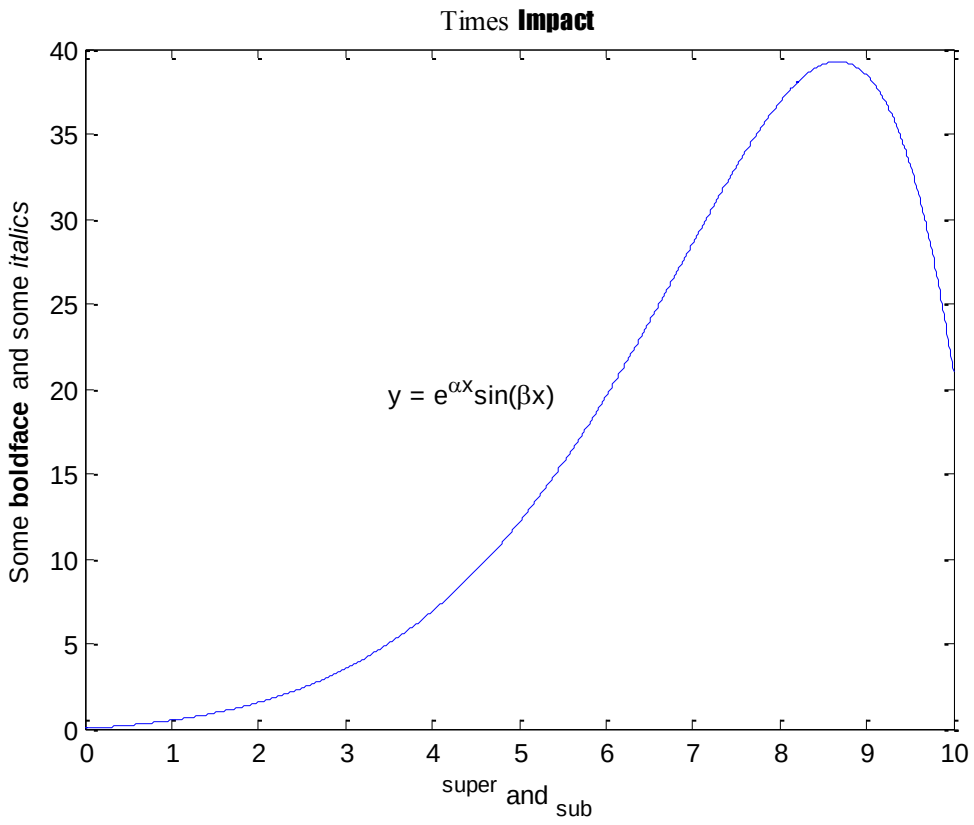
ملاحظة: يتيح لنا MATLAB إمكانيات كبيرة من أجل إضافة ملاحظات وتعليقات للأشكال كأسماء المحاور والأكال، عرضنا بعض التعليمات في الأمثلة السابقة، نعرض في هذه الملاحظة إمكانيات إضافية هي:

- كتابة أحرف إغريقية.
- التحكم بحجم وشكل التسميات.

مثال لتوضيح الفكرة السابقة:

قم بكتابة الرماز التالي:

```
%%
clear all;
close all;
clc;
alpha = 0.5;
beta = 0.3;
x = 0:.01:10;
y = exp(alpha*x).*sin(beta*x);
figure();
plot(x,y);
title('\fontname{Times}{Times} \fontname{Impact}{Impact}')
xlabel('^{super} and _{sub}')
ylabel('Some \bfboldface\rm and some \it{italics}')
text(3.5, 20, 'y = e^{\alphax}sin(\betax)')
```

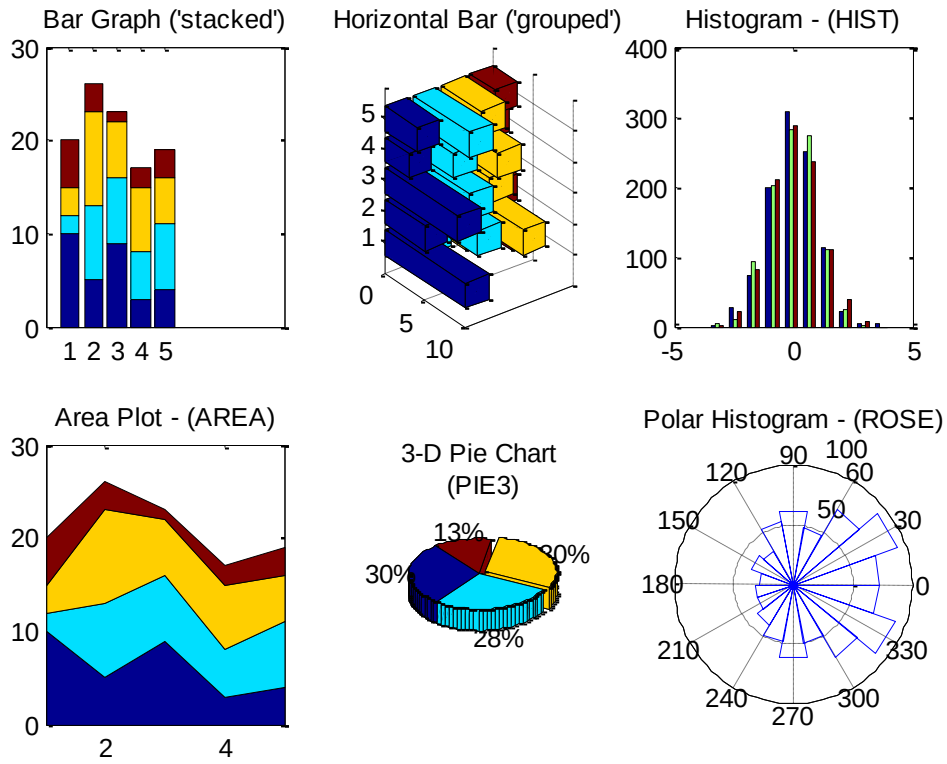


ملاحظة 2: يتيح لنا MATLAB أيضاً إمكانيات كبيرة وخيارات واسعة من أجل إظهار الأشكال حيث تم تعريف ضمن مكاتب البرمجية عدد كبير من أنواع الرسوم لإظهار الأشكال سنستعرض العديد منها في هذا المثال، حيث نرغب في جعل هذه الرمازات البرمجية أساس يعود لها الطالب عند الحاجة لاستخدام نوع معين من الرسوم، يجب قراءة كل رماز بشكل دقيق:

```
%%
clear all;
close all;
clc;
data = [10 2 3 5; 5 8 10 3; 9 7 6 1; 3 5 7 2; 4 7 5 3];
subplot(2,3,1)
bar(data, 'stacked');
title('Bar Graph (''stacked'')');
subplot(2,3,2)
bar3h(data);
title('Horizontal Bar (''grouped'')');
```

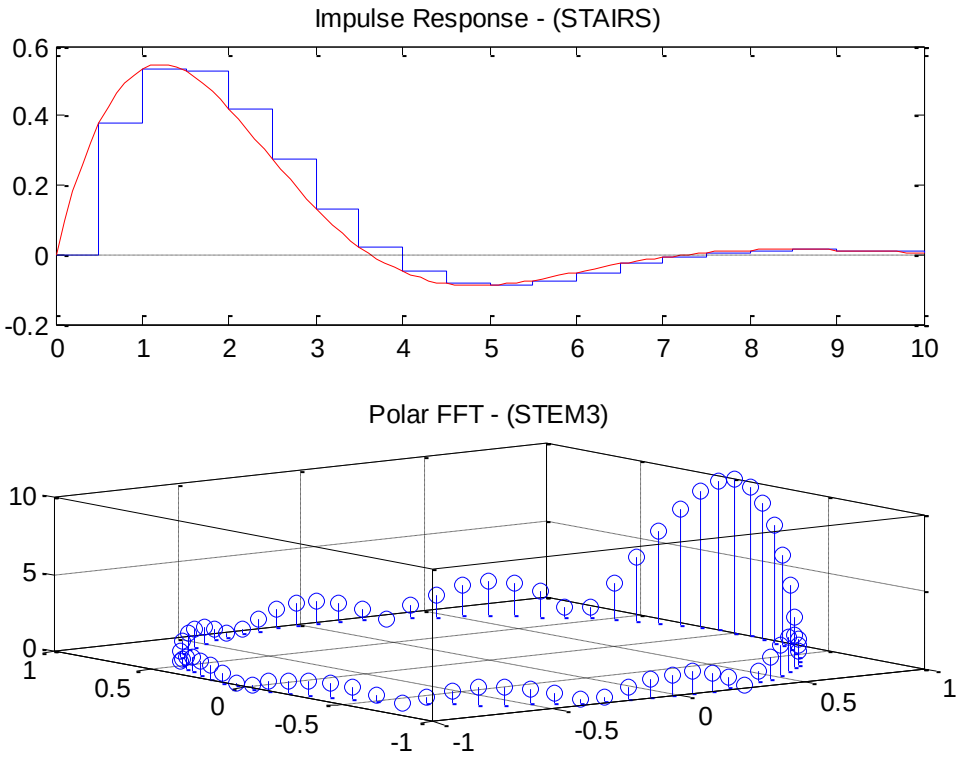
```
subplot(2,3,3)
hist(randn(1000,3));
title('Histogram - (HIST)');
subplot(2,3,4)
area(data);
title('Area Plot - (AREA)');
subplot(2,3,5)
pie3(sum(data), [0 0 1 0]);
title(['3-D Pie Chart';'    (PIE3)   ']);
subplot(2,3,6)
rose(5/3*randn(1000,1), 18);
title('Polar Histogram - (ROSE)');
```

يظهر الشكل التالي:



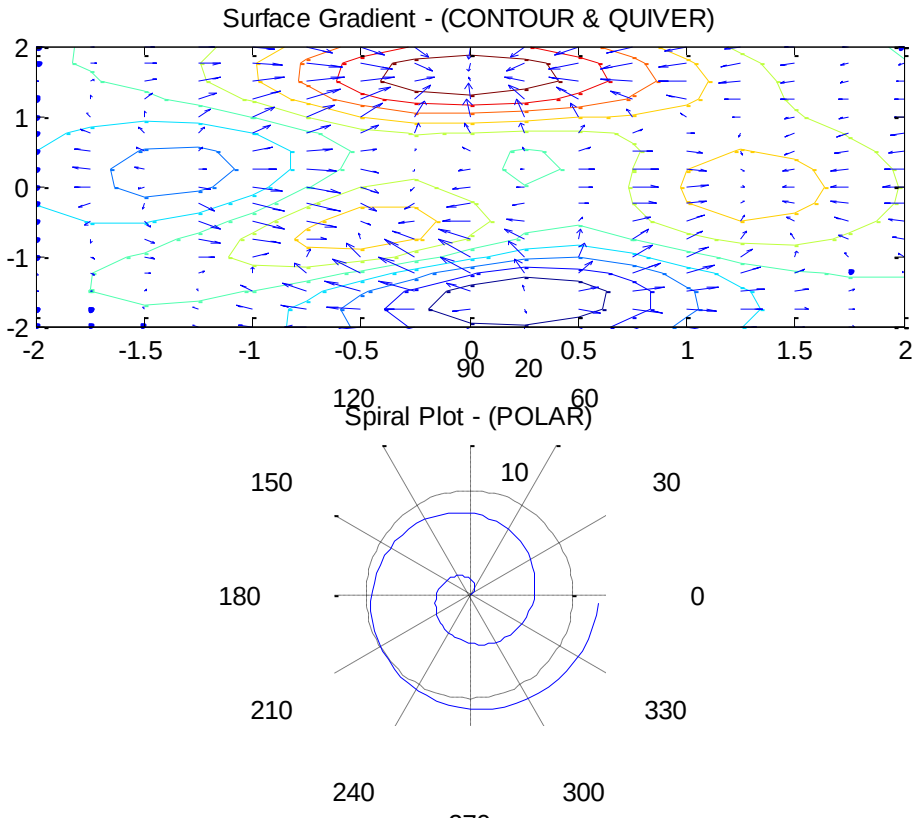

```
clear all
```

```
close all;
clc;
subplot(2,1,1)
t = 0:0.1:10;
z = impulse(1, [1 1 1], t);
stairs(t(1:5:end),z(1:5:end))
hold on
plot(t,z,'r')
plot([0 t(end)], [0 0], 'k:')
title('Impulse Response - (STAIRS)')
subplot(2,1,2)
theta = 2*pi*(0:74)/75;
x = cos(theta);
y = sin(theta);
z = abs(fft(ones(10,1), 75))';
stem3(x, y, z)
title('Polar FFT - (STEM3)')
```




```
clear all
close all;
clc;
subplot(2,1,1)
[X,Y,Z] = peaks(-2:0.25:2);
[U,V] = gradient(Z, 0.25);
contour(X,Y,Z,10);
hold on
quiver(X,Y,U,V);
title('Surface Gradient - (CONTOUR & QUIVER)')
theta = 0:0.1:4*pi;
[x,y] = pol2cart(theta(1:5:end), theta(1:5:end));
subplot(2,1,2)
polar(theta,theta)
axis([-13 13 -12.5 14.5])
```

```
title('Spiral Plot - (POLAR)')
```

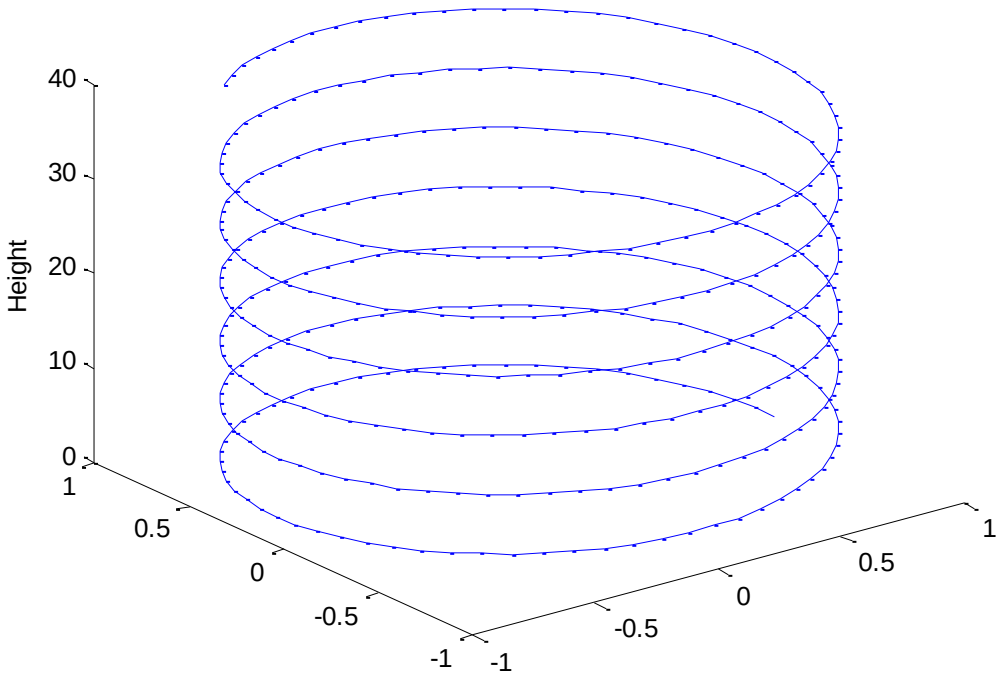


مثال 3D Plot:

يمكن رسم توابع ثلاثية الأبعاد استخدام plot3، سنوضح في هذا الرماز استخدام هذه التعليمة:

```
%%
 clear all
close all;
clc;
z=0:.1:40;
x=cos(z);
y=sin(z);
grid;
plot3(x,y,z);
zlabel('Height');
```

سنحصل على الشكل التالي:



يمكن إضافة التعليمة rotate3d من أجل الإتاحة للمستخدم إمكانية تدوير الشكل.

مثال 3D Surface Plotting:

سنعرض في هذه الأمثلة خيارات الإظهار المتعلقة برسم السطوح لأشكال ثلاثية الأبعاد.

فيما يلي بعض التعليمات المفيدة:

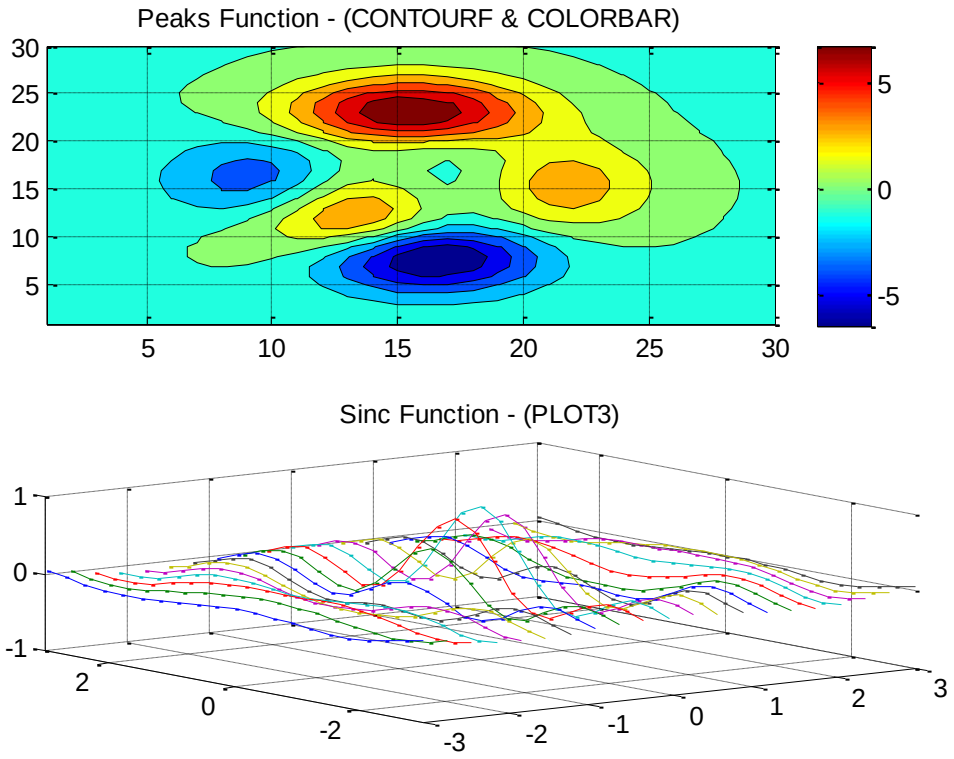
- **contour** - 2D contour lines
- **contourf** - 2D filled contour lines
- **contour3** - 3D contour lines
- **plot3** - 3D lines of columns of data
- **mesh** - 3D wire-frame view of the surface
- **surf** - 3D perspective plot of the surface
- **waterfall** - 3D lines of rows of data with a reference planer

مثال 1:

```
%%
clear all
close all;
clc;
subplot(2,1,1)
```

```
x = -3:0.3:3; y = x;
[X,Y]=meshgrid(x,y);
[theat,R] = cart2pol(X,Y);
Z = sinc(R);
contourf(peaks(30), 10)
colorbar
grid on
title('Peaks Function - (CONTOURF & COLORBAR)')

subplot(2,1,2)
plot3(X,Y,Z)
grid on
axis([-3 3 -3 3 -1 1])
title('Sinc Function - (PLOT3)')
```

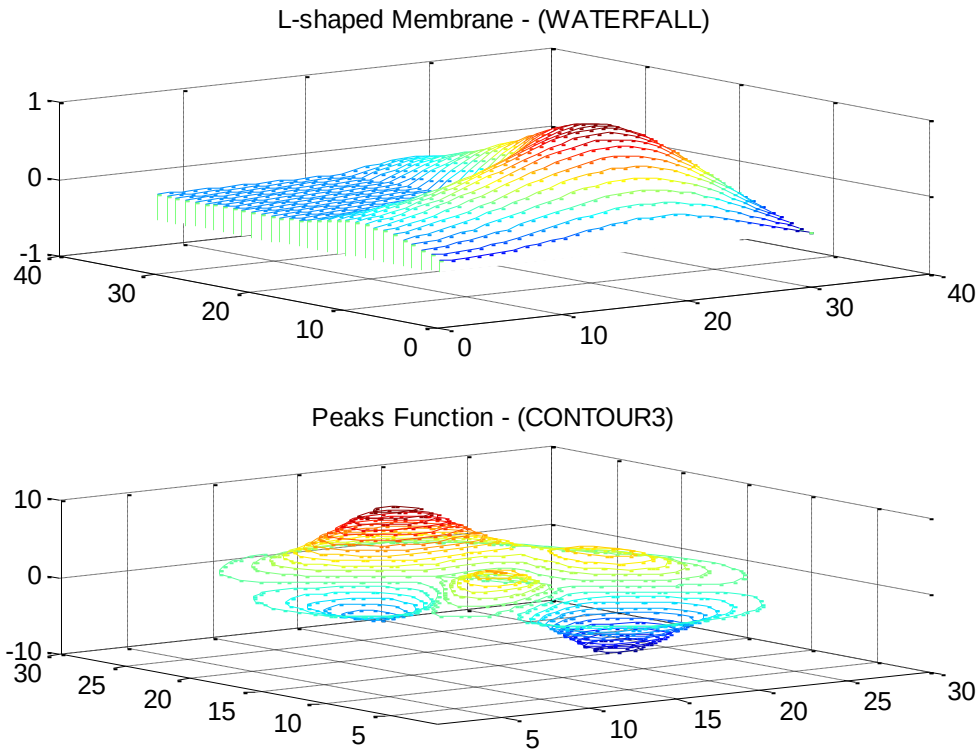
الخرج بعد التنفيذ:



مثال 2:

```
%%
clear all
close all;
clc;
subplot(2,1,1)
waterfall(membrane(1));
title('L-shaped Membrane - (WATERFALL)')

subplot(2,1,2)
contour3(peaks(30), 25);
title('Peaks Function - (CONTOUR3)')
```

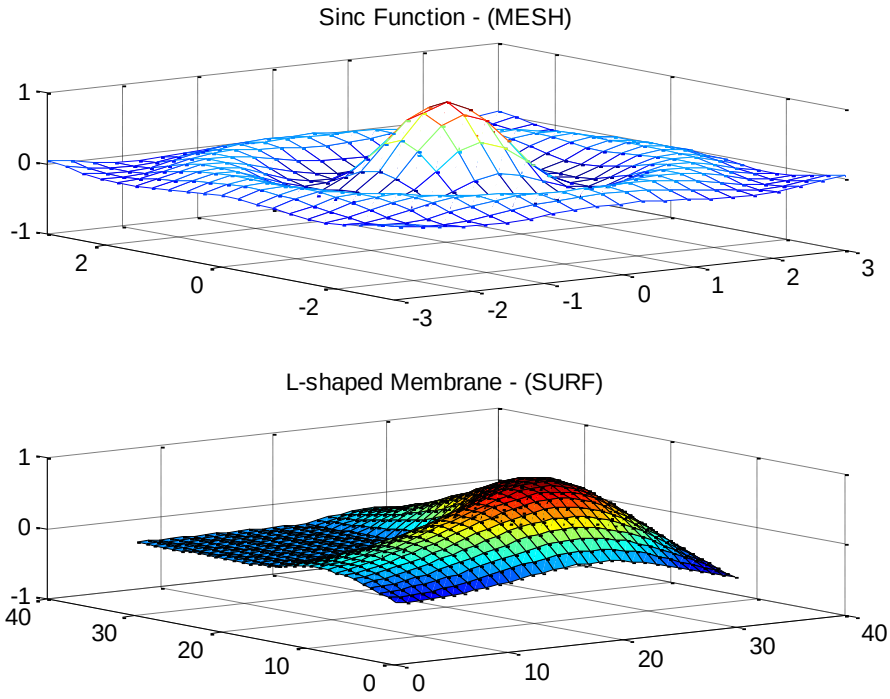
الخرج بعد التنفيذ:



مثال 3:

```
%%
clear all
close all;
clc;
x = -3:0.3:3; y = x;
[X,Y]=meshgrid(x,y);
[theat,R] = cart2pol(X,Y);
Z = sinc(R);
subplot(2,1,1)
mesh(X,Y,Z)
axis([-3 3 -3 3 -1 1])
title('Sinc Function - (MESH)')

subplot(2,1,2)
surf(membrane(1))
title('L-shaped Membrane - (SURF)')
```

الخرج على الشكل التالي:


Sinc Function - (MESH)
1
0
-1
2
0
-2
-3
-2
-1
0
1
2
3
L-shaped Membrane - (SURF)
1
0
-1
40
30
20
10
0
0
10
20
30
40

الأسئلة

1. اذكر خمسة من عبارات التحكم بالتدفق

مساعدة: قراءة فقرة تعابيير التحكم بالتدفق

2. قم بكتابة تابع وسمه checkvariable يقوم بفحص قيمة المتحول وإعادة رسالة على الشكل التالي:
إذا كان المتحول أكبر من 5 أو أصغر من 20- يعيد متحول يحتوي الرسالة "this variable is invalid"
أما إذا كانت القيمة غير ذلك أي أصغر من 5 وأكبر من 20- سوف يعيد متحول يحوي الرسالة التالية "this variable is valid"

مساعدة: قراءة فقرة عبارة if كاملةً

3. قم بكتابة تابع وسمه checkdays يؤخذ التابع دخل له اسم من أسماء الاسبوع وحسب الاسبوع يعيد ترتيبه أي
الأحد يظهر الرقم 1 ..... السبت يظهر الرقم 7

مساعدة: قراءة فقرة عبارة switch

4. ضمن ملف script قم بكتابة رماز برمجي يقوم بإظهار كل قوى عدد 2 ضمن المجال [256 – 2].

مساعدة: قراءة فقرة حلقة while

5. ضمن ملف script قم بكتابة رماز برمجي يقوم بإظهار كل الإعداد الزوجية ضمن المجال [30 – 60] .
ثم أنشئ خلية جديدة وقم بإظهار كافة الأعداد الفردية ضمن المجال [40 – 10].

مساعدة: قراءة فقرة حلقة for

6. قم بتعريف متحولين من نمط string الأول يحتوي الكلمة "TEST" والثاني "Chapter 4" ومن ثم قم بتعريف متحول يقوم بضم المتحولين السابقين بشكل أفقي.

مساعدة: قراءة فقرة السلاسل المحرفية

7. قم برسم التابع اللوغاريتمي على المجال 0.1:100 بخطوة 0.1 وذلك باسخدام التعليمات plot, stem و stairs و semilogx
قم بتسمية محور الزمن ب Time ومحور الترتيب ب Log(t) إضافةً لوضع عنوان للرسمة وهو Plot Log Function وتغيير لون المنحني إلى اللون الأسود

مساعدة: قراءة فقرة البيانيات والإظهار

الإجابات

1.

if/elseif/else
for
switch/case/otherwise
try/catch
while
break
continue
end
pause
return

2.

```
function y = checkvariable( x )
 if x < -20
y='this variable is invalid';
 elseif x > 5
 y='this variable is invalid';
  else
  y='this variable is valid';
  end
  End
```

.3
```
function [ ord_day ] = checkdays( day )
switch day
 case{ 'Sunday'}
ord_day=1;
 case{ 'Monday'}
 ord_day=2;
 case{ 'Tuesday'}
 ord_day=3;
 case{ 'Wednesday'}
  ord_day=4;
 case{ 'Thursday'}
  ord_day=5;
  case{ 'Friday'}
   ord_day=6;
   case{ 'Saturday'}
    ord_day=7;
     disp(ord_day)
      end
      End
```

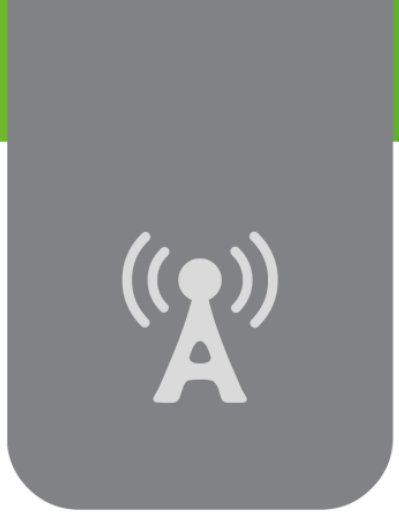
.4
```
power = 2;
disp(power)
while power < 256
 power =power* 2;
 disp(power);
 end
```

.5
```
for i = 30 : 2 : 60
disp(i);
End
```

```
%%
for i = 11 : 2 : 40
disp(i);
End
```

.6
```
str1='TEST '
str2='Chapter 4'
c= [str1,' ',str2]
```

.7
```
t=0.1:0.1:100;
y=log(t);
plot(t,y,'k');
xlabel('Time');
ylabel('Log(t)');
title('Plot Log Function');
stem(t,y)
stairs(t,y);
semilogx(t,y);
```



الفصل الخامس: تمثيل المعطيات (استيراد وتصدير المعطيات، التعامل مع الصور والملفات الصوتية، مقدمة عن الأدوات)

Data Representations (Data Import & Export, Dealing with Images & Sounds, Toolboxes)

عنوان الموضوع:

تمثيل المعطيات (استيراد وتصدير المعطيات، التعامل مع الصور والملفات الصوتية، مقدمة عن الأدوات)
Data Representations (Data Import & Export, Dealing with Images & Sounds, Toolboxes)

الكلمات المفتاحية:

MATLAB، أنماط المعطيات Data type، سلاسل محرفية string، محارف characters، أعداد صحيحة integer، double، فاصلة عائمة floating point، تحويل conversion، ملفات دخل / خرج Input/Output file or I/O file، صور images، ملفات صوتية sound files، الأدوات toolboxes.

ملخص:

نهتم في هذا الفصل بالتعرف على أنماط المعطيات، وبشكل خاص الصفوف الموجودة ضمن MATLAB من أجل تمثيل هذه المعطيات، ثم ننتقل للتعرف على توابع MATLAB التي تؤمن لنا تحديد نمط المعطيات أو تتيح لنا التحويل من نمط إلى آخر مما قد يفيد في حفظ الذاكرة أو تقليل زمن تنفيذ الرماز المكتوب بلغة MATLAB.
ننتقل بعدها للتعرف على التوابع عالية أو منخفضة المستوى والتي تفيد في التعامل مع ملفات دخل / خرج، حيث سيتم قراءة المعطيات من ملف نصي، وتعلم الكتابة بملف نصي، وسيتم أيضاً التعامل مع معطيات مخزنة ضمن ملفات برنامج Excel.
نتعرف أيضاً ضمن هذا الفصل على كيفية التعامل مع الصور والملفات الصوتية ضمن MATLAB. ننتقل في الجزء الأخير للتعرف على الأدوات Toolboxes المتوفرة ضمن برمجية MATLAB.

أهداف تعليمية:

يتعرف الطالب في هذا الفصل على:

- التعامل مع مختلف أنماط المعطيات.
- إتقان وفهم تحديد أنماط المعطيات والتعرّف عليها وتحديدها.
- استخدام توابع عالية المستوى او منخفضة المستوى في التعامل مع الملفات.
- استخدام توابع MATLAB لاستيراد وتصدير المعطيات وتخزينها من وإلى ملفات نصية، أو ملفات Excel مثلاً.
- التعامل مع الملفات الصوتية والصور ضمن MATLAB.
- التعرف على الأدوات Toolboxes ضمن MATLAB وتطبيقاتها المتنوعة.

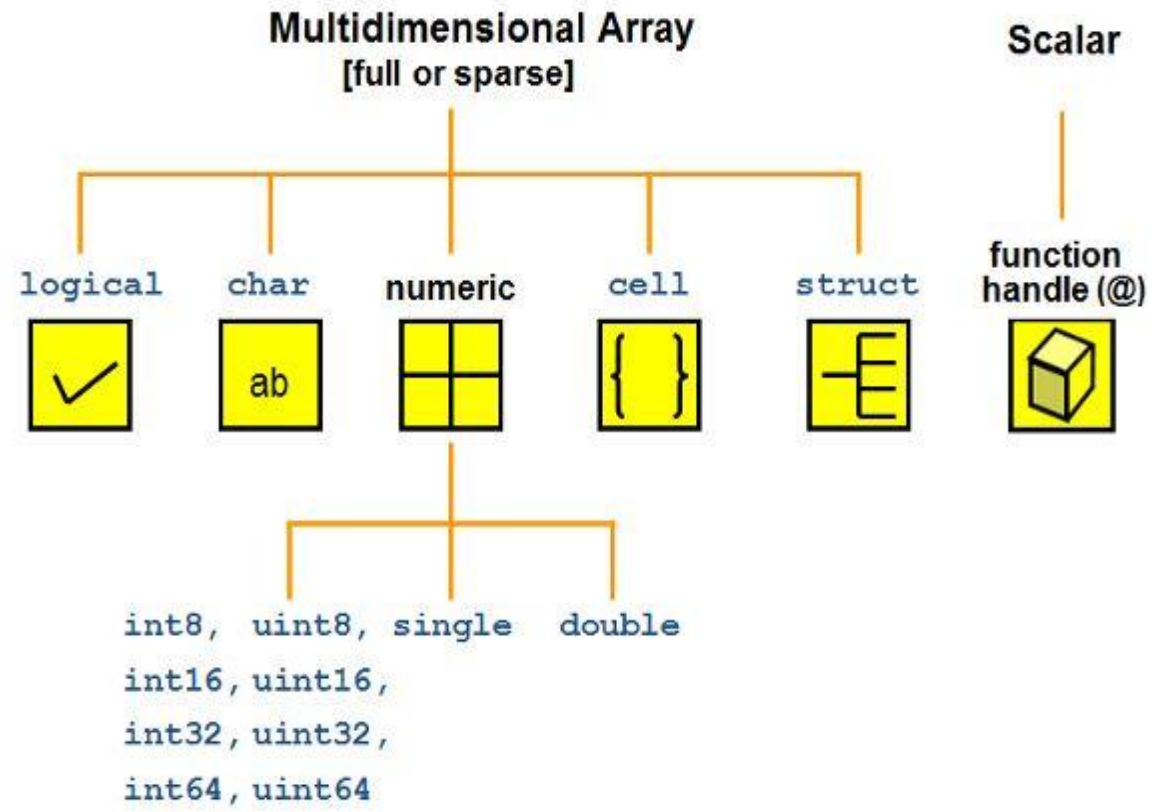
Data types أنماط المعطيات

قدمنا ضمن فقرة (العمل مع المتحولات ضمن MATLAB في الفصل الثاني من هذا المقرر) شرحاً عن أنماط المعطيات والصفوف المتوفرة ضمن MATLAB، وبعد التعامل مع MATLAB ضمن الفصول السابقة أصبحت الميزة التالية واضحة لأي مستخدم للبرمجية، وهي أنه لا حاجة لتعريف نمط أو تحديد أبعاد أي متحول ضمن MALTAB، حيث يقوم المستخدم بتعريف المتحول الجديد حصراً عن طريق تعريف اسمه وإسناد قيمة له، ثم يتم إسناد نمط وحجز موقع له تلقائياً في الذاكرة، ومن الممكن للمستخدم التعديل على قيم المتحول أو إضافة قيم جديدة كأن يكون المتحول عنصراً سلميّاً ثم يصبح شعاعاً من القيم السلمية.

سنقدم الآن تذكرة سريعة بهذه الأنماط ثم سننتقل إلى فقرتين مهمتين:

- التعرّف على(تحديد) أنماط المعطيات Data Taype Indentification.
- التحويل بين أنماط المتحولات Data Type Conversion.

يبين الشكل التالي الصفوف الرئيسية الموجودة ضمن MATLAB.



يوجد ضمن برمجية MATLAB العديد من انماط المعطيات المتنوعة، أو ما يعرف بالصفوف Classes، والتي يستطيع المبرمج استخدامها، حيث يمكن بناء مصفوفات وصفيفات من معطيات صحيحة integer، باستخدام الفاصلة العائمة floating point، محارف characters، سلاسل محرفية strings، أو متحولات منطقية (true or false). إضافةً لما سبق يمكن استخدام struct أو خلية صفيفات cell arrays من أجل تخزين المعطيات المتباينة (غير المتشابهة) مثلاً اسم الموظف، رقمه، عمره، أيام عمله.

التعرّف على (تحديد) أنماط المعطيات Data Type Identification

يتضمن MATLAB على العديد من التوابع المبنية والتي تفيد المستخدم في تحديد نمط المعطيات لمتحول ما، نعرض في الجدول التالي هذه التوابع إضافةً إلى الغرض purpose من استخدامها:

Function	Purpose
is	Detect state
isa	Determine if input is object of specified class
iscell	Determine whether input is cell array
iscellstr	Determine whether input is cell array of strings
ischar	Determine whether item is character array
isfield	Determine whether input is structure array field
isfloat	Determine if input is floating-point array
ishghandle	True for Handle Graphics object handles
isinteger	Determine if input is integer array
isjava	Determine if input is Java object
islogical	Determine if input is logical array
isnumeric	Determine if input is numeric array
isobject	Determine if input is MATLAB object
isreal	Check if input is real array
isscalar	Determine whether input is scalar
isstr	Determine whether input is character array
isstruct	Determine whether input is structure array
isvector	Determine whether input is vector
class	Determine class of object
validateattributes	Check validity of array
whos	List variables in workspace, with sizes and types

يجب على الطالب قراءة التوابع السابقة ومعرفة عملها بشكل دقيق فهي مفيدة جداً في الكثير من التطبيقات مثلاً عند تطوير برامج، أو بناء واجهات تخاطبية أو حتى بناء توابع تحل مشكلة معينة، فمثلاً من الضروري فحص الدخل قبل البدء بالعمليات.

مثال: الهدف هو استخدام بعض من التوابع السابقة. قم بكتابة الرماز التالي، وفسر نتيجة خرجه.

```
x =3;

isinteger(x)
```

```
isfloat(x)
isvector(x)
isscalar(x)
isnumeric(x)
y =23.54;
isinteger(y)
isfloat(y)
isvector(y)
isscalar(y)
isnumeric(y)
```

مثال:

لنقم ببناء تابع يقوم بتحديد عدد الأيام ضمن شهر ما يقوم المستخدم بإدخاله، وفقاً لما يلي:

- الدخل هو عبارة عن اسم الشهر.
- يجب استخدام تعليمة switch.
- القيمة المعادة من التابع هي عدد أيام الشهر.
- يعيد التابع رسالة خطأ في الحالات التالية:
  - الدخل ليس كلمة.
  - الدخل ليس اسم لشهر من أشهر السنة.

الحل: يبين الرماز التالي التابع المبني للقيام بالوظيفة السابقة.

```
function [ days ] = days_in_month( month )
% Returns the number of days for a given month
if ~ischar(month)
   days=sprintf('Error: you did not enter the name of a month');
   return
end
switch month
   case{ 'january' 'march' 'may' 'july' 'august' 'october' 'december'}
      days=31;
   case { 'april' 'june' 'september' 'november' }
```

```
    days =30;
  case {'february' }
    days=28;
  otherwise
    days=sprintf('%s is not the name of a month',month);
    disp(days)
end

end
```

بعد فهم التابع، قم بإنشاء ملف script واختبر التابع السابق.
نلاحظ استخدام التابع ischar للتأكد من كون الدخل هو محرف.
ينصح دوماً باستخدام هذه التوابع عند الحاجة وذلك من أجل بناء توابع او برامج مهمة وذات تطبيق عملي بحيث تناقش جميع الحالات للدخل، كما ينصح دوماً باستخدامها مع تعليمة if.

التحويل بين أنماط المعطيات Data Type Conversion

يحتوي MATLAB على العديد من التوابع المبنية والتي تفيد المستخدم في التحويل بين نمط معطيات ما ونمط آخر، نعرض فيما يلي التوابع المفيدة في عملية التحويل هذه:

Function	Purpose
char	Convert to character array (string)
int2str	Convert integer data to string

mat2str	Convert matrix to string
num2str	Convert number to string
str2double	Convert string to double-precision value
str2num	Convert string to number
native2unicode	Convert numeric bytes to Unicode characters
unicode2native	Convert Unicode characters to numeric bytes
base2dec	Convert base N number string to decimal number
bin2dec	Convert binary number string to decimal number
dec2base	Convert decimal to base N number in string
dec2bin	Convert decimal to binary number in string
dec2hex	Convert decimal to hexadecimal number in string

hex2dec	Convert hexadecimal number string to decimal number
hex2num	Convert hexadecimal number string to double-precision number
num2hex	Convert singles and doubles to IEEE hexadecimal strings
cell2mat	Convert cell array to numeric array
cell2struct	Convert cell array to structure array
cellstr	Create cell array of strings from character array
mat2cell	Convert array to cell array with potentially different sized cells
num2cell	Convert array to cell array with consistently sized cells
struct2cell	Convert structure to cell array

تفيد هذه التوابع كثيراً عند بناء الواجهات التخاطبية، يجب الاطلاع عليها بشكل مفصل إضافةً للشرح والأمثلة ضمن Help وذلك وفقاً للمسار التالي:

MATLAB → Language Fundamentals → Data Types → Data Type conversion → converting from Numeric to String

و

MATLAB → Language Fundamentals → Data Types → Data Type conversion → converting from String to Numeric

استيراد المعطيات Data Import

يعني استيراد المعطيات ضمن MATLAB جلبها من ملف خارجي، مثلاً ملف text. يمكن باستخدام التابع **importdata** استيراد أنواع مختلفة من ملفات المعطيات وبعدة أشكال وصيغ، نعرض فيما يلي الصيغ التي تؤخذها التعليمة:

صيغة التابع	الوصف
A = importdata(filename)	تقوم هذه التعليمة بجلب المعطيات وتخزينها ضمن الصفيفة A من ملف اسمه filename، أي الاسم الذي يقوم المستخدم بتحديده. الخرج هو عبارة مصفوفة أو صفيفة متعدّدة الأبعاد حسب طبيعة الملف. يفضّل أيضاً ذكر لاحقة الملف.
A = importdata('-pastespecial')	تقوم هذه التعليمة بجلب المعطيات من ذاكرة التخزين المؤقتة عوضاً عن الملف.
A = importdata(____ , delimiterIn)	إن delimiterIn هو عبارة المحرف الخاص لفصل الأعمدة ضمن ملف ASCII، أو الملف الذي يقوم المستخدم بتطبيق التعليمة عليه، أو الذاكرة المؤقتة. يمكن استخدام delimiterIn مع الشكلين القواعديين السابقين. يتم تحديده كسلسلة محرفية string، أمثلة: ',' أو ' '
A = importdata(____ , delimiterIn, headerlinesIn)	جلب المعطيات من ملف ASCII، أو الملف الذي يقوم المستخدم بتطبيق التعليمة عليه، أو الذاكرة المؤقتة ويتم قراءة المعطيات ابتداءً من: **headerlinesIn+1**
[A, delimiterOut, headerlinesOut] = importdata(filename)	يمكن عند كتابة التعليمة بالشكل التالي الحصول على المعطيات مخرنة في A إضافةً للحصول على محرف delimiter المستخدم ضمن الملف للفصل بين عناصر و الحصول على عدد سطور رأس الملف header ضمن الملف.

ملاحظة مهمة جداً جداً جداً:

يوجد ملفات مرفقة مع الفصل الحالي، عبارة عن ملفات معطيات وصور يرجى وضعها ضمن مسار العمل الحالي من أجل تطبيق التعليمات التالية وإلا سيعطي التنفيذ خطأ.

مثال 1:

سنقوم باستيراد وعرض صورة، مخزنة باسم ولاحقة Koala.png ومن ثم صورة Desert.jpg

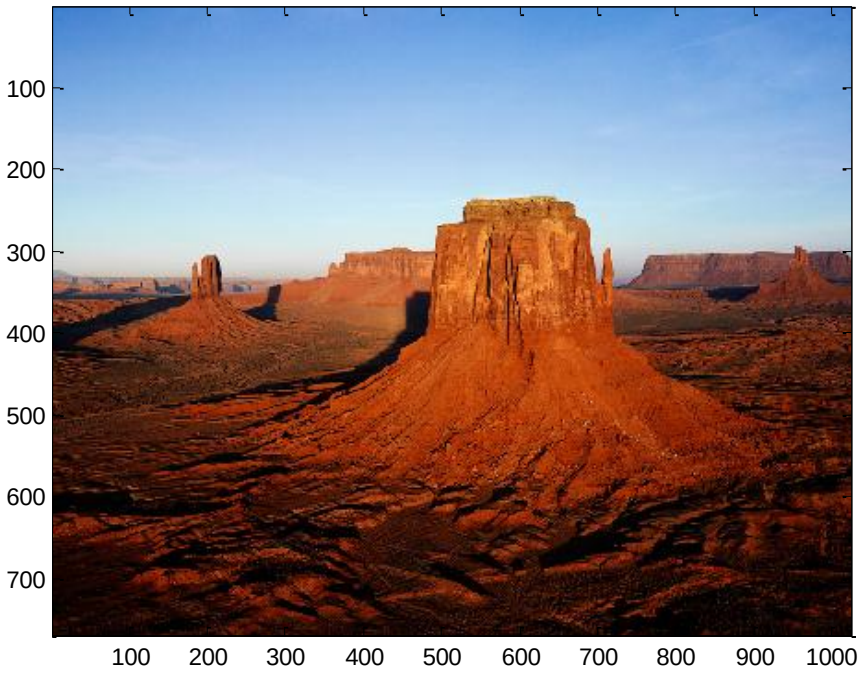
Editor - C:\Users\Simon\Documents\MATLAB\imp_data.m

imp_data.m

```
1      %% Import data
2      % image
3 -    filename ='Koala.png';
4 -    A = importdata(filename);
5 -    image(A)
6 -    filename ='Desert.jpg ';
7 -    A1 = importdata(filename);
8 -    image(A1)
9
```

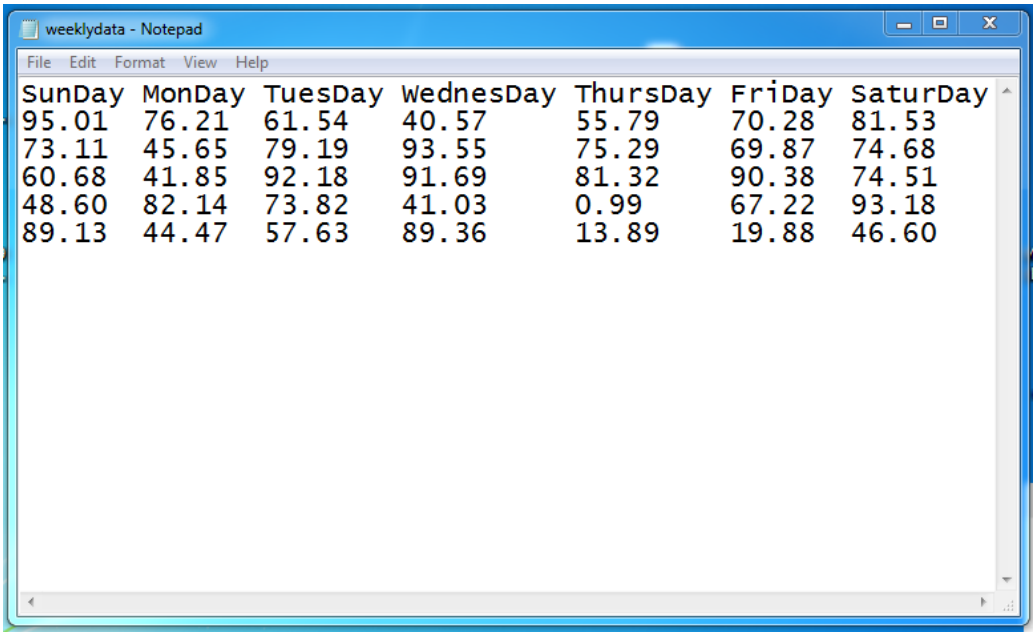
عند تنفيذ أول ثلاثة أسطر نحصل على الخرج التالي:

ثم بعد تنفيذ آخر ثلاثة أسطر نحصل على



ملاحظة: قم بتجريب الرماز السابق على كافة الصور الموجودة في الملف المرفق.

مثال: نرغب في هذا المثال في استيراد ملف نصي باسم weeklydata.txt وسيتم تحديد Delimiter إضافةً إلى Column Header.يوضح الشكل التالي المعطيات الموجودة ضمن الملف.



نقوم بكتابة الرماز التالي:

Editor - C:\Users\Simon\Documents\MATLAB\imp_data.m

imp_data.m

```
%% Import .txt file
clear all;close all;clc;
filename ='weeklydata.txt';% file we want to import it
delimiterIn =' ';% delimiter is string and equal to space
headerlinesIn =1;% header is only one row
A = importdata(filename,delimiterIn,headerlinesIn);
%View data
for k =1:7
   disp(A.colheaders{1, k})
   disp(A.data(:, k))
   disp(' ')
end
```

الخرج على الشكل التالي:

Command Window

```
SunDay
   95.0100
   73.1100
   60.6800
   48.6000
   89.1300


MonDay
   76.2100
   45.6500
   41.8500
   82.1400
   44.4700


TuesDay
   61.5400
   79.1900
   92.1800
   73.8200
   57.6300
```

مثال: سنقوم في هذا المثال باستيراد معطيات من الذاكرة المؤقتة.

ضمن ملف script مثلاً ثم اكتب التعليمة التالية:

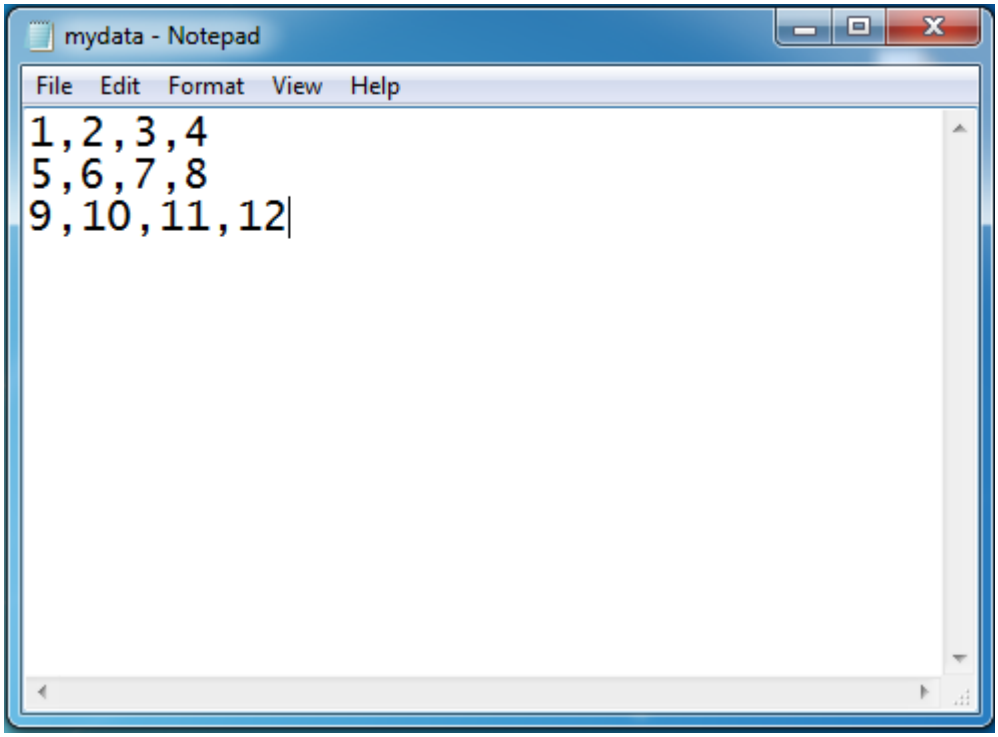
```
A = importdata('-pastespecial')
```

ثم اكتب عبارة I love MATLAB course !

ونفذ التعليمة السابقة فقط ولاحظ الخرج.

أو أنشئ ملف نصي واكتب به بعض الكلمان ثم نفذ التعليمة ولاحظ الخرج ضمن MATLAB.

مثال: أنشئ ملف نصي باسم mydata.txt على الشكل التالي:



ثم قم بكتابة الرماز التالي:

```
%%
clear all;close all;clc;
filename = 'mydata.txt';
[A,delimiterOut]=importdata(filename)
```

الخرج هو:

نلاحظ أن MATLAB قام بمعرفة delimiter تلقائياً.

يمكن إضافةً لما سبق التعامل مع المتحولات الموجودة ضمن فضاء العمل Worksapce، فمثلاً من المفيد أحياناً حفظ قيم المتحولات ضمن ملف ما لإعادة استخدامه في وقت لاحق.

يتيح لنا MATLAB العديد من الخيارات للتعامل مع المتحولات ضمن فضاء العمل مثل استيرد المتحول، حفظه وحذفه load, save and delete، حيث يتم حفظ هذه المتحولات إلى MAT-file بلاحقة (.mat).

من أجل حفظ المتحول أو جميع المتحولات ضمن فضاء العمل يمكن من خلال Toolstrip ضمن جزء Home، ضمن قسم Variable استخدام الخيار Save Worksapce كما في الشكل.

أو يمكن الضغط على المتحول ضمن فضاء العمل ثم الضغط على الزر اليميني للفأرة يظهر عدة خيارات، يمكن اختيار save as، او استخدام التوابع مثل تابع save.

من أجل استيراد المتحول يمكن استخدام التابع load حيث يستخدم هذا التابع بشكل عام من اجل جلب المعطيات المخزنة بملف mat. إلى فضاء العمل، أما من أجل مسح المتحولات فكما تعلمنا مسبقاً باستخدام clear.

مثال:

سنقوم باستيراد ملف cloudwater.dat ومن ثم غظهار النتائج، يحتوي هذا الملف على قيم مخزنة مسبقاً لأحد المتحولات ضمن Worspace ونرغب في رسمها، يجب التأكد من وضع الملف ضمن مسار العمل.

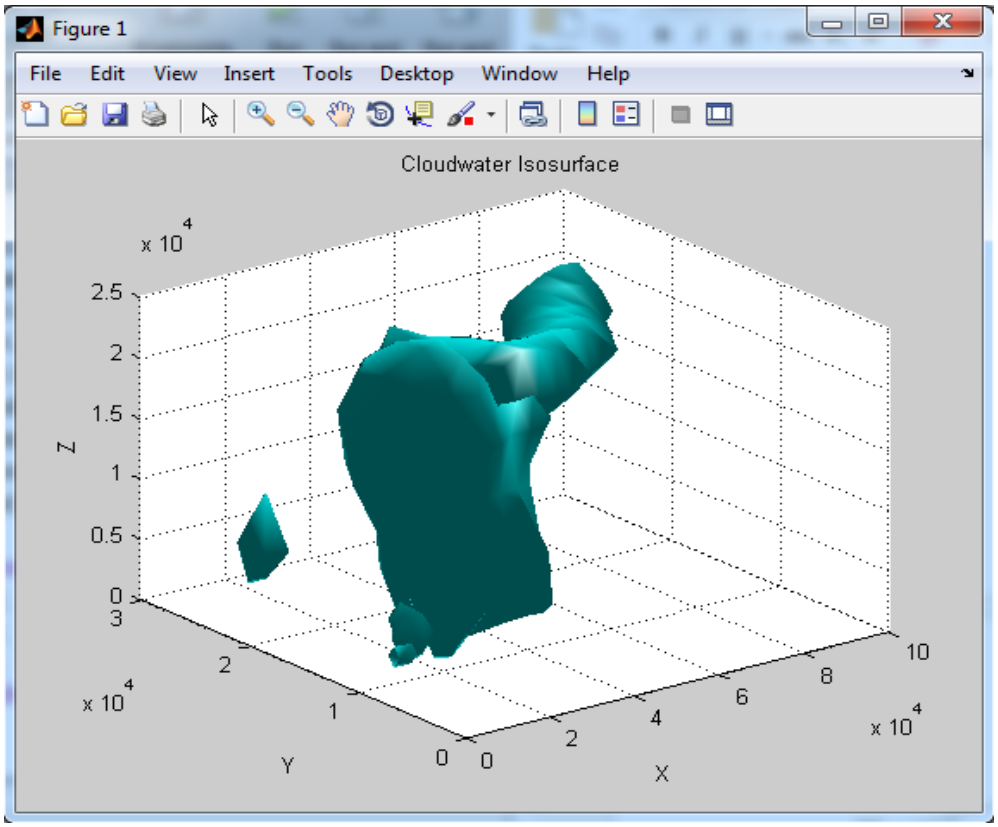
ومن ثم سنقوم ببعض العمليات على المعطيات من أجل إظهار الشكل المهم هو فهم كيفية استيراد ملف المعطيات اما العمليات فهي ليست ضرورية الآن.

Editor - C:\Users\Simon\Documents\MATLAB\cloudwater.m

cloudwater.m

```
clear all;close all;clc;
d=importdata('cloudwater.dat');
x1=0:4166.6666:100000;
y1=0:2214.286:28785.718;
z1=0:4153.846:29076.922;
[x,y,z]=meshgrid(x1,y1,z1);
d2=shiftdim(reshape(d,[8 14 25]),1);
p=patch(isosurface(x,y,z,d2,.12),'FaceColor','cyan','EdgeColor','none');
isonormals(x,y,z,d2,p);
camlight;
lighting gouraud
xlabel('X');
ylabel('Y');
zlabel('Z');
title('Cloudwater Isosurface');
view(3);
grid on;
```

الخرج هو:



مثال: سنقوم في هذا المثال باستيراد معطيات من ملف excel بلاحقة csv..

يقوم البرنامج الذي سنقوم بكتابته بتقديم بعض المعلومات الإحصائية لمجموعة من معطيات الأسعار اليومية لأحد المخازن وذلك ضمن فترة معينة ممتدة بين عام 2005 و 2015 حيث يقوم بإيجاد المتوسط للأسعار لكل السنوات إضافةً للمتوسط لكل سنة على حدى، المعطيات باسم sample_data.csv لا تنسى وضعها ضمن مسار العمل.

قم بكتابة الرماز التالي:

Editor - C:\Users\Simon\Documents\MATLAB\csv_file.m

csv_file.m

```
1     %% Import Data from Excel file
2     % load chosen data to MATLAB
3 -   clear all;close all;clc;
4 -   stocks= csvread('sample_data.csv');
5     % MATLAB convert .csv file data into matrices
6 -   mean_st=mean (stocks);
7 -   for i = 0 : 9
8 -      mean(stocks(250*i+1:250*i+250) )
9 -   end
```

ولاحظ الخرج.

- التعامل مع ملفات دخل/ خرج بتعليمات منخفضة المستوى Low-Level File I/O

وجدنا في الشرح السابق أن استخدام تابع importdata يتيح لنا استيراد المعطيات، إلا أن هذا التابع يعمل على مستوى مرتفع High Level. تتيح لنا توابع منخفضة المستوى Low Level المخصصة للتعامل مع ملفات دخل/ خرج تحكم أكبر وخيارات أرسع في قراءة أو كتابة المعطيات إلى ملف، إلا أنها تحتاج إلى معلومات أكثر عن الملف المطلوب العمل عليه وذلك لضمان عملها بشكل فعّال.

يؤمن MATLAB عدد من التوابع لإجراء عمليات القراءة أو الكتابة على مستوى البايت byte أو المحرف character، يعرض الجدول التالي هذه التوابع:

Function	Description
Fclose	Close one or all open files
Feof	Test for end-of-file
Ferror	Information about file I/O errors
Fgetl	Read line from file, removing newline characters
Fgets	Read line from file, keeping newline characters
Fopen	Open file, or obtain information about open files
Fprintf	Write data to text file
Fread	Read data from binary file
Frewind	Move file position indicator to beginning of open file
Fscanf	Read data from text file
Fseek	Move to specified position in file
Ftell	Position in open file
Fwrite	Write data to binary file

لنتعلم الآن كيفية استيراد المعطيات من ملف نصي باستخدام تعليمات دخل/ خرج منخفضة المستوى Low-Level I/O

يتم عادةً الاستعانة بالتوابع التالية للتعامل مع الملفات النصية:

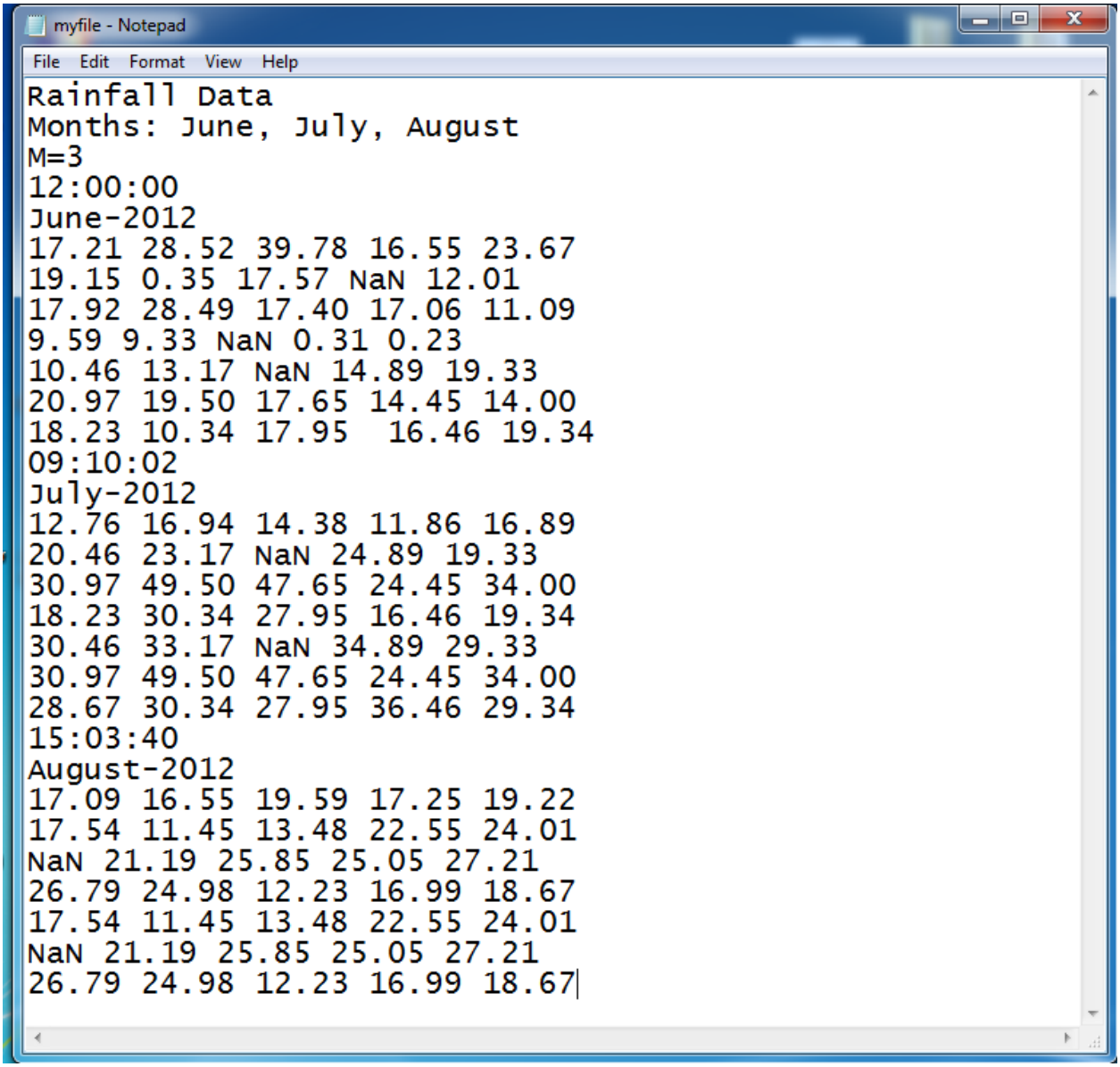
- تابع fscanf يقوم بقراءة المعطيات الموجودة في ملف نصي أو ملف ASCII.
- تابعي fgetl و fgets تقرأ سطر واحد من الملف عند تنفيذها لمرة واحدة. والفرق بين التابعين هو في المحافظة على أحرف السطر الجديد أم عدمها.
- تابع fread يقوم بقراءة دفقة من المعطيات على مستوى البايت byte أو البت bit.

مثال:

لدينا ملف معطيات باسم myfile.txt تم حفظه ضمن مسار العمل، تم تخزين ضمن هذا الملف معطيات عن سقوط الأمطال لثلاثة أشهر هي June و July و August في عام 2012.

إن المعطيات المخزنة ضمن الملف myfile.txt تحتوي مجموعة مكرّرة من الزمن، الشهر، وقياسات هطول الأمطال في خمسة اماكن، بادئة الملف header يحتوي على عدد الأشهر M لذلك سيكون لدينا M مجموعة من القياسات.

المعطيات موضحة في الشكل التالي:



سنقوم باستيراد المعطيات من الملف ومن ثم عرض هذه المعطيات، وذلك باتباع الخطوات التالية:

1. فتح الملف باستخدام تعليمة fopen والحصول على معرف الملف file IDentifier.
2. وصيف المعطيات ضمن الملف باستخدام format specifiers مثل '%s' للسلاسل المحرفية string، '%d' للأعداد الصحيحة integer، و'%f' للأعداد الممثلة بالفاصلة العائمة floating point.

3. من أجل إهمال أحرف ضمن الملف، يجب تضمنيهم ضمن format description الخاص بالتعليمة fscanf حيث يجب استخدام (*) ثم نمط أو صيغة المعطيات التي ترغب بإهمالها مثلاً للحصول على قيمة M المخزنة ضمن الملف استخدام الرماز التالي:

```
M = fscanf(fid,'%*s %*s\n%*s %*s %*s %*s\nM=%d\n\n',1);
```

4. تقوم تعليمة fscanf تلقائياً بقراءة المعطيات وذلك تبعاً لتوصيف الصيغ format description الذي قمنا بوضعه إلى حين لم تعد قادرة على الموافقة بين المعطيات والتوصيف الذي قمنا بكتابته، سنقوم في هذا الرماز باستخدام حلقة for من أجل قراءة ثلاث مجموعات من المعطيات كل مرة، وسيتم قراءة سبعة أسطر و خمس أعمدة.
5. سيتم إنشاء struct يدعى Mydata من أجل حفظ المعطيات، التي تم قرائتها من الملف، ضمنه ضمن فضاء العمل workspace. تحتوي هذه البنية struct على ثلاثة حقول هي time, month and raindata aaray.

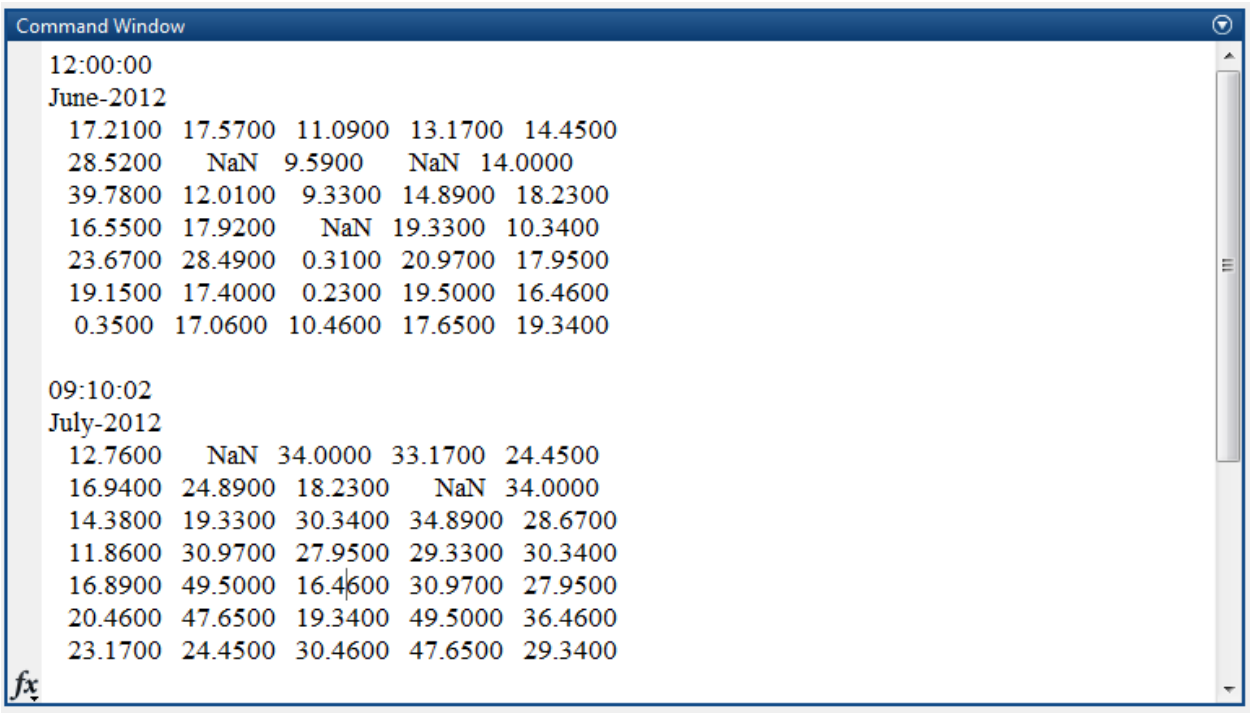
قم بكتابة الرماز التالي، ولا تنسى بوضع ملف المعطيات ضمن مسار العمل:

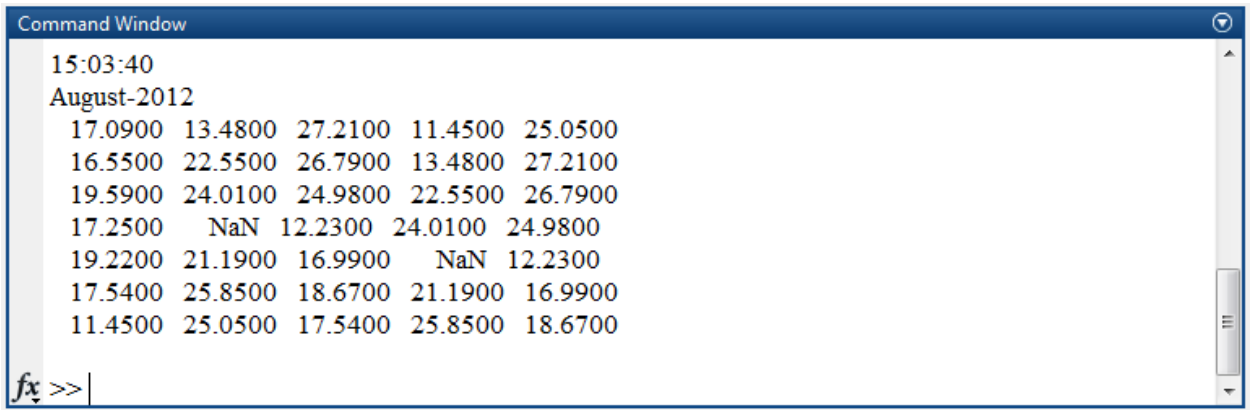
Editor - C:\Users\Simon\Documents\MATLAB\file_LowLevel.m

file_LowLevel.m

```
filename ='myfile.txt';
rows =7;
cols =5;
% open the file
fid = fopen(filename);
% read the file headers, find M (number of months)
M = fscanf(fid,'%*s %*s\n%*s %*s %*s %*s\nM=%d\n\n',1);
% read each set of measurements
for n =1:M
   mydata(n).time = fscanf(fid,'%s',1);
   mydata(n).month = fscanf(fid,'%s',1);
   % fscanf fills the array in column order,
   % so transpose the results
   mydata(n).raindata =...
      fscanf(fid,'%f',[rows, cols]);
end
for n =1:M
   disp(mydata(n).time), disp(mydata(n).month)
   disp(mydata(n).raindata)
end
% close the file
fclose(fid);
```

الخرج على الشكل التالي:





تصدير المعطيات Data Export

نقصد بتصدير المعطيات data export هو كتابة المعطيات ضمن ملفات. يتيح لنا MATLAB استخدام المعطيات ضمن تطبيقات أخرى تستطيع قراءة ملفات ASCII، هذه الطرق عديدة سنركز في هذه الفقرة على تصدير المعطيات إلى ملف نصي باستخدام تعليمات Low-Level I/O.

مثال: اكتب الرماز التالي

```
Editor - C:\Users\Simon\Documents\MATLAB\exp_data.m
exp_data.m
 1      %%%
 2      % create a matrix y, with two rows
 3 -    clear all;close all;clc;
 4 -    x = 0:10:100;
 5 -    y = [x  ;log(x)];
 6      % open a file for writing
 7 -    fid = fopen('logtable.txt', 'w');
 8      % Table Header
 9 -    fprintf(fid, 'Log      Function\n\n');
10      % print values in column order
11      % two values appear on each row of the file
12 -    fprintf(fid, '%f   %f\n', y);
13 -    fclose(fid);
14      % display the file created type logtable.txt
```

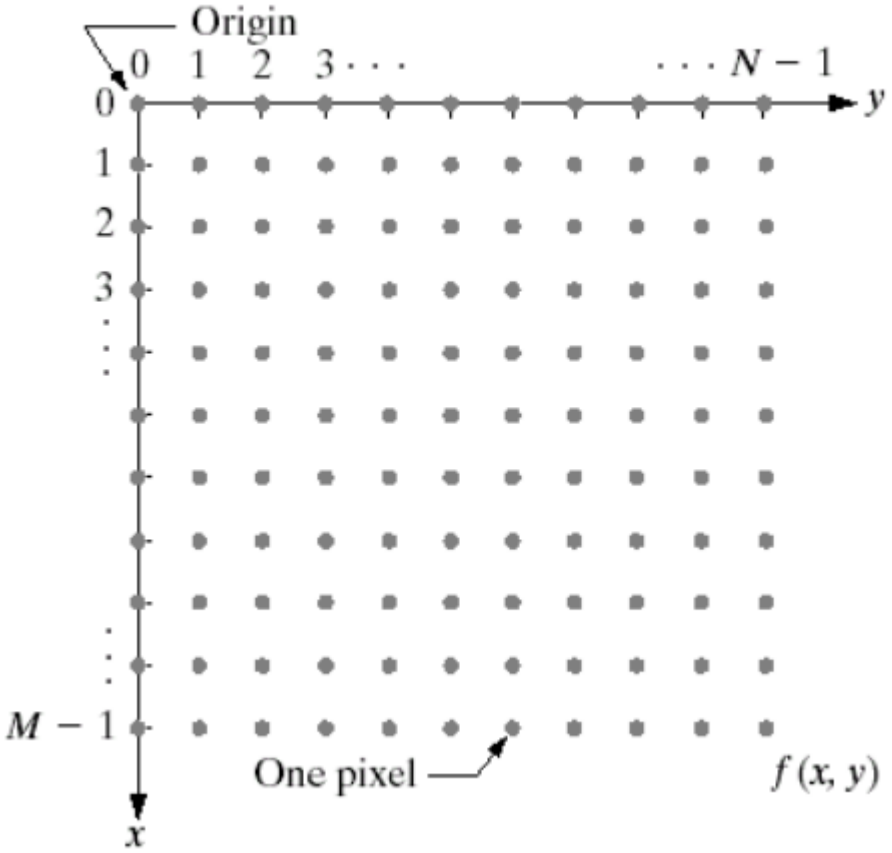
الخرج على الشكل التالي وهو عبارة عن ملف نصي.

```
C:\Users\Simon\Documents\MATLAB\logtable.txt - No...
File  Edit  Search  View  Encoding  Language  Settings  Macro
Run  Plugins  Window  ?                                    X
logtable.txt
 1  Log        Function
 2
 3  0.000000   -Inf
 4  10.000000   2.302585
 5  20.000000   2.995732
 6  30.000000   3.401197
 7  40.000000   3.688879
 8  50.000000   3.912023
 9  60.000000   4.094345
10  70.000000   4.248495
11  80.000000   4.382027
12  90.000000   4.499810
13  100.000000   4.605170
14
Ln:14  Col:1  Sel:0|0  UNIX  ANSI  INS
```

التعامل مع الصور Dealing with Images

نحتاج في عدد كبير من التطبيقات إلى التعامل مع الصور، يؤمن لنا MATLAB عدد كبير من التوابع التي تتعامل مع الصور بهدف تطبيق مفاهيم معالجة الصورة image precessing، سنركز في ذه الفقرة فقط على عد تتعليمات من أجل قراءة الصورة وإظهارها.

يمكن النظر إلى أي صورة كتابع لمتحولين $f(x,y)$ له بعدين x,y ويعبر مطاله عن شدة الإضاءة من أجل النقطة (x,y). حيث تتكون الصورة من مجموعة عناصر منتهية ندعوها pixel، وبالتالي يمكن تمثيل المصفوفة على شكل مصفوفة $M \times N$ كما في الشكل التالي:



يوجد أنواع عديدة من الصور نذكر منها: PNG, JPEG, GIF, TIFF, BMP حيث تختلف الصور حسب طريقة تخزينها وترميزها.

سنستخدم التوابع التالية للتعامل مع الصور:

- imread لقراءة الصورة.
- imshow لإظهار الصورة.
- imwrite لكتابة الصورة ضمن ملف جديد.
- iminfo لعرض معلومات عن الصورة.

مثال: سنقوم في هذا المثال بقراءة صورة، ثم إظهارها ومعرفة بعض المعلومات عنها، ثم كتابة هذه الصورة ضمن ملف جديد اسمه yourname.jpeg أي تغيير لاحقة الصورة.

Editor - C:\Users\Simon\Documents\MATLAB\image_file.m

image_file.m

```
%% Image Processing
clear all;close all;clc;
Baby_image=imread('baby.jpg'); % read the image
imshow('baby.jpg');% show the image
fo=imfinfo('baby.jpg');% get information
imwrite(Baby_image,'yourname.jpeg'); % write the image into a new file
```

الخرج:

سيتم إنشاء struct ضمن فضاء العمل قم بفتحه لمعرفة معلومات غضافية عن الصورة، وتفقد مسار العمل ولاحظ إنشاء صورة جديدة باسم ولاحقة yourname.jpeg.

التعامل مع الملفات الصوتية Dealing with Sounds

يعتبر التعامل مع الملفات جزءاً هاماً في معظم تطبيقات الحياة اليومية وخاصة نقل الصوت أو التعديل عليه أو كشف بعض الخواص الإحصائية المميزة للصوت، فمثلاً يمكن لنا معرفة التردّد الأساسي لصوتنا ومن الممكن تطبيق خوارزميات للتعديل على النغمات والأصوات ومن ثم تطبيق مفاهيم معالجة الإشارة الكلامية لتحسين جودة الصوت، ومن الممكن ايضاً تطبيق خوارزميات ضغط الصوت المستخدمة في الأجهزة الخلوية مثلاً.

مثال: سنقوم بتسجيل صوت عن المايك وذلك باستخدام الرماز التالي، يطلب عند بدأ التسجيل أن تتكلم من اجل سماع صوتك بعد تسجيل المقطع.

Editor - C:\Users\Simon\Documents\MATLAB\sound_file.m

sound_file.m

```
1       %% Sound Processing
2       % Record The Sound
3 -     clear all; close all; clc;
4 -     recObj = audiorecorder;
5 -     disp('Start speaking.')
6 -     recordblocking(recObj, 5);% record for 5 second
7 -     disp('End of Recording.');
8 -     play(recObj);
```

يتم التسجيل لمدة خمسة ثواني.

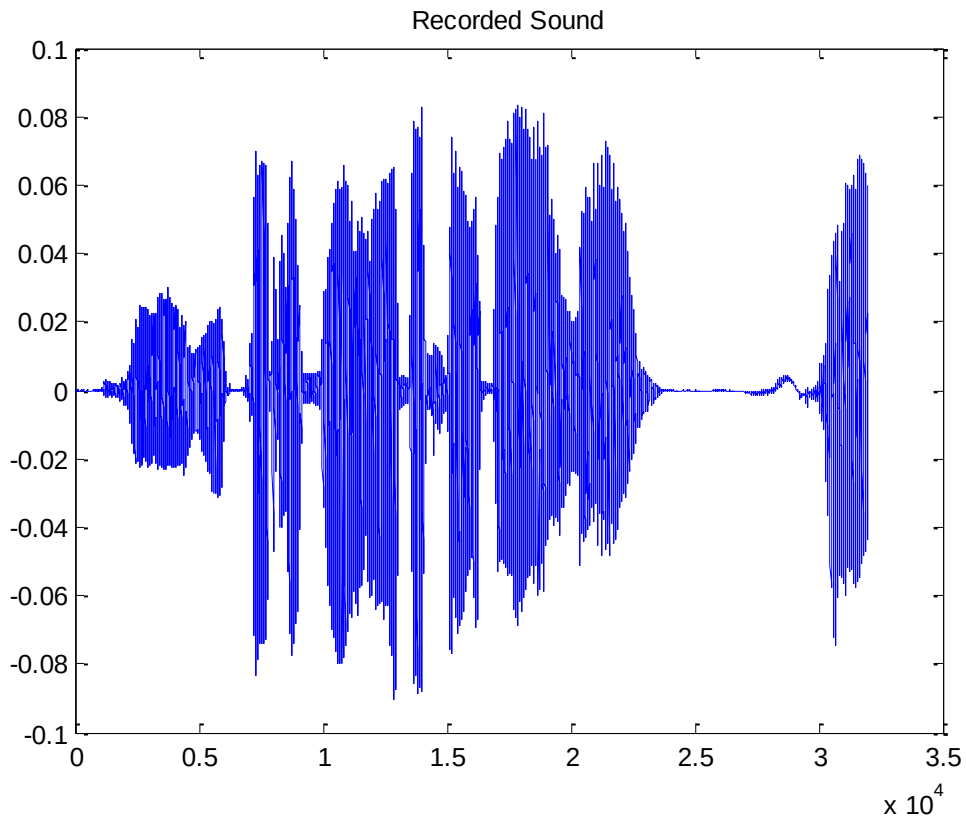
مثال2 : نعرض في هذا المثال تعليمات لتسجيل ملف صوتي من نوع wav هذه التعليمات لم تعد مستخدمة في النسخ الحديثة من MATLAB وقد تم استبدالها بالتعليمات السابقة، الهدف هو فقط الاطلاع عليها.

Editor - C:\Users\Simon\Documents\MATLAB\sound_file.m

sound_file.m

```
10      % Record The Sound
11 -    Fs=8e3; % Sampling Frequency
12 -    npoints=4*Fs; % number of points for wavrecord
13 -    sound_rec=wavrecord(npoints,Fs);% record for 4 second
14 -    wavplay(20*sound_rec,Fs);% play the sound
15 -    figure();
16 -    plot(sound_rec);% Plot the Sound Signal
17 -    title('Recorded Sound');
```

نقوم بتسجيل المقطع الصوتي، ثم نقوم بتشغيله بعد تضخيمه 20 مرة ومن ثم نقوم برسمه.



الأدوات Toolboxes

على الرغم من أن الهدف الأساسي لبرمجية MATLAB هو إجراء الحسابات العدّدية، إلا أن MATLAB تقدم مجموعة متنوعة من الإضافات extensions والمكاتب libraries الخاصة بالبرمجية، وذلك على غرار مختلف لغات البرمجة، هذه الإضافات مفيدة في طيف واسع من التطبيقات.

قام المطورون والقائمون على إعداد وإخراج البرمجية، بتجميع عدد كبير وشامل من المكاتب libraries والتوابع functions المتخصصة بتطبيق أو مجال معيّن، وتطويرها وإخراجها على شكل أداة Toolbox تفيد المستخدم في إيجاد حلول لتطبيق محدّد application-specific solutions أو صف خاص من المشكلات، هذه المكاتب السابقة تدعى الأدوات Toolboxes.

Toolboxes are comprehensive collections of libraries, MATLAB functions (M-files), and Routines that are designed to provide application-specific solutions and solve particular classes of problems

يعتبر استخدام toolboxes أمراً مهماً لغالبية مستخدمي MATLAB، وذلك لأنها تتيح لك **تعلم** و **تطبيق** تقانات وخوارزميات في مجالات واختصاصات مختلفة، تغطي Toolboxes ضمن MATLAB مجالاً واسعاً من التطبيقات ابتداءً من تقديم وسيلة أبسط عن طريق برامج لإجراء حسابات اعتيادية، كحساب مقلوب مصفوفة مثلاً أو ترشيح إشارة بمرشح ما، مروراً بتطبيق خوارزميات معينة في مجال ما على الدخل مباشرةً دون الخوض في تفاصيل تنجيز الخوارزمية، مثلاً تطبيق خوارزميات معالجة الصور على الصور كإيجاد الحواف أو كشف الوجوه، وانتهاءً بالحصول على نتائج، في التنبؤ أو الاستيفاء مثلاً، لا يمكنك الحصول عليها مباشرةً دون هذه الأدوات.

Toolboxes will make your life easier because you don't have to write them, and test them

نذكر بعض من المجالات والاختصاصات التي تشتمل عليها MATLAB Toolboxes:

معالجة الإشارة والصورة

أنظمة التحكم

الشبكات العصبونية Neural Networks

الاتصالات

الرياضيات والإحصاء والأمثلة Optimization.

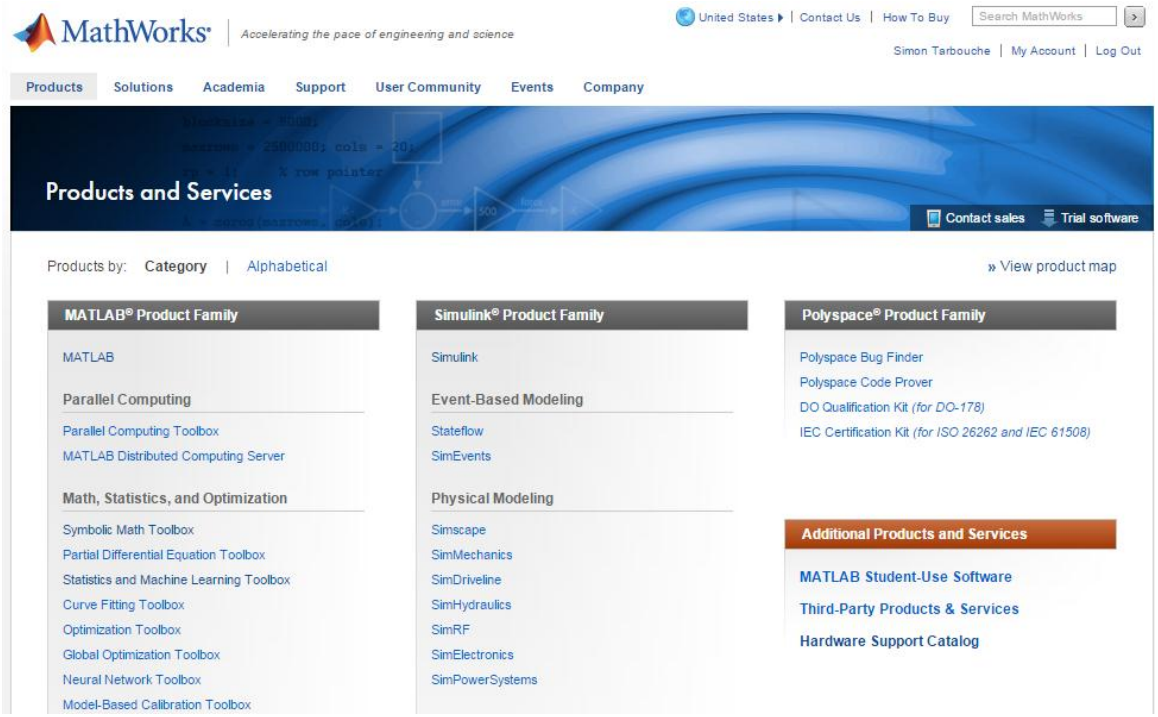
يمكن الاطلاع على جميع toolboxes المتوفرة ضمن برمجية MATLAB، عن طريق زيارة الموقع الالكتروني لشركة MathWorks® والدخول إلى Products.

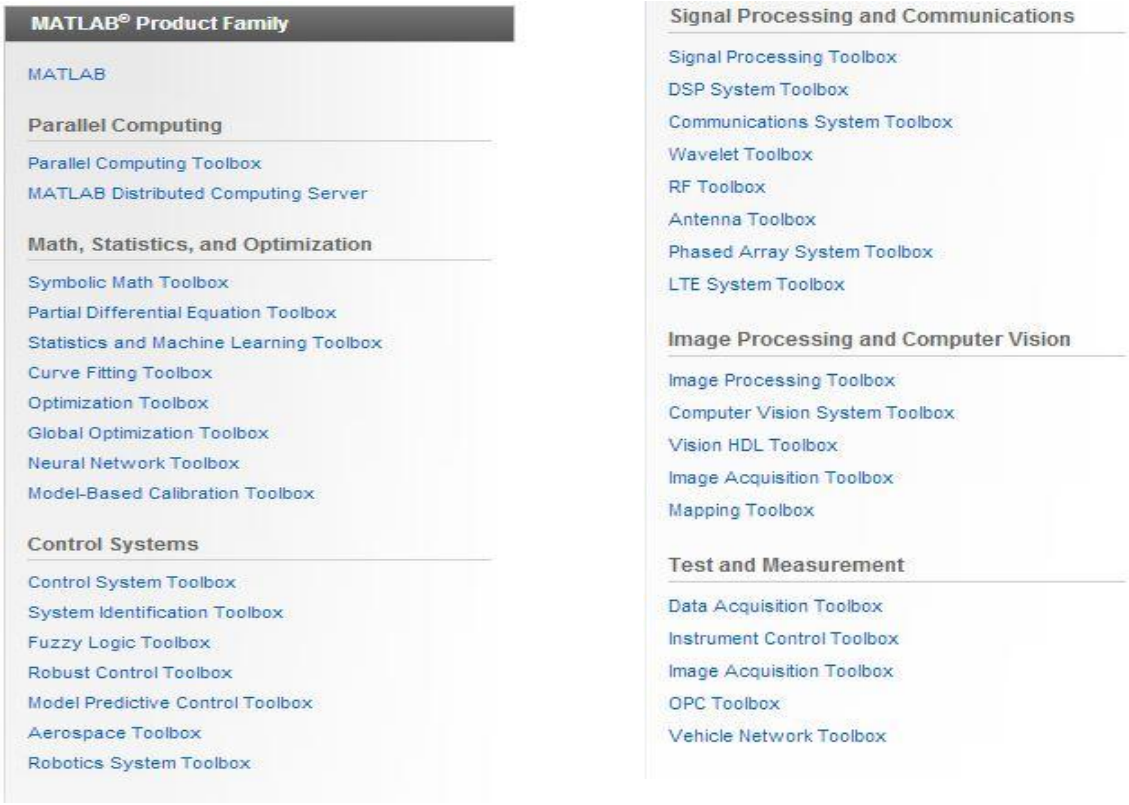
http://www.mathworks.com/

الصور التالية تعرض بعض المنتجات والخدمات المتعلقة ب MATLAB وإضافةً إلى الأدوات Toolboxes التي تؤمنها شركة MathWorks® وذلك ضمن مجالات مختلفة وتطبيقات متعدّدة، حيث تسعى شركة

MathWorks® إلى إضافة Toolboxes جديدة مع كل إصدار جديد لبرمجية MATLAB أو إضافة تحسينات لجميع Toolboxes الموجودة مسبقاً.

http://www.mathworks.com/products/?s_tid=gn_ps





Computational Finance

Financial Toolbox
Econometrics Toolbox
Datafeed Toolbox
Database Toolbox
Spreadsheet Link EX *(for Microsoft Excel)*
Financial Instruments Toolbox
Trading Toolbox

Computational Biology

Bioinformatics Toolbox
SimBiology

Code Generation and Verification

MATLAB Coder
HDL Coder
Vision HDL Toolbox
HDL Verifier
Filter Design HDL Coder
Fixed-Point Designer

Application Deployment

MATLAB Compiler
MATLAB Compiler SDK
Spreadsheet Link EX *(for Microsoft Excel)*
MATLAB Production Server

Database Connectivity and Reporting

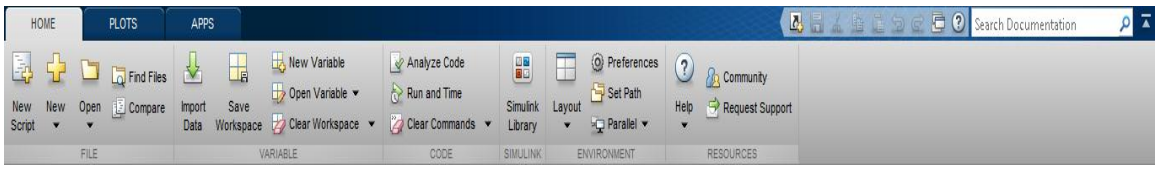
Database Toolbox
MATLAB Report Generator

Signal Processing and Communications

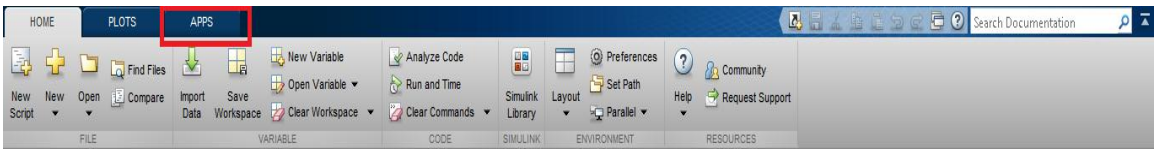
DSP System Toolbox
Communications System Toolbox
SimRF
Computer Vision System Toolbox

إن جميع Toolboxes السابقة جرى مكاملتها مع بيئة MATLAB ضمن الواجهة الرئيسية، يمكن معرفة جميع الـ toolboxes المرفقة مع النسخة الخاصة بك من MATLAB عن طريق التعليمة **ver**، يمكن كتابتها ضمن Command Window لمعرفة المعلومات السابقة.

من أجل الوصول إلى toolboxes ضمن البرمجية، يمكن استخدام Toolstrip



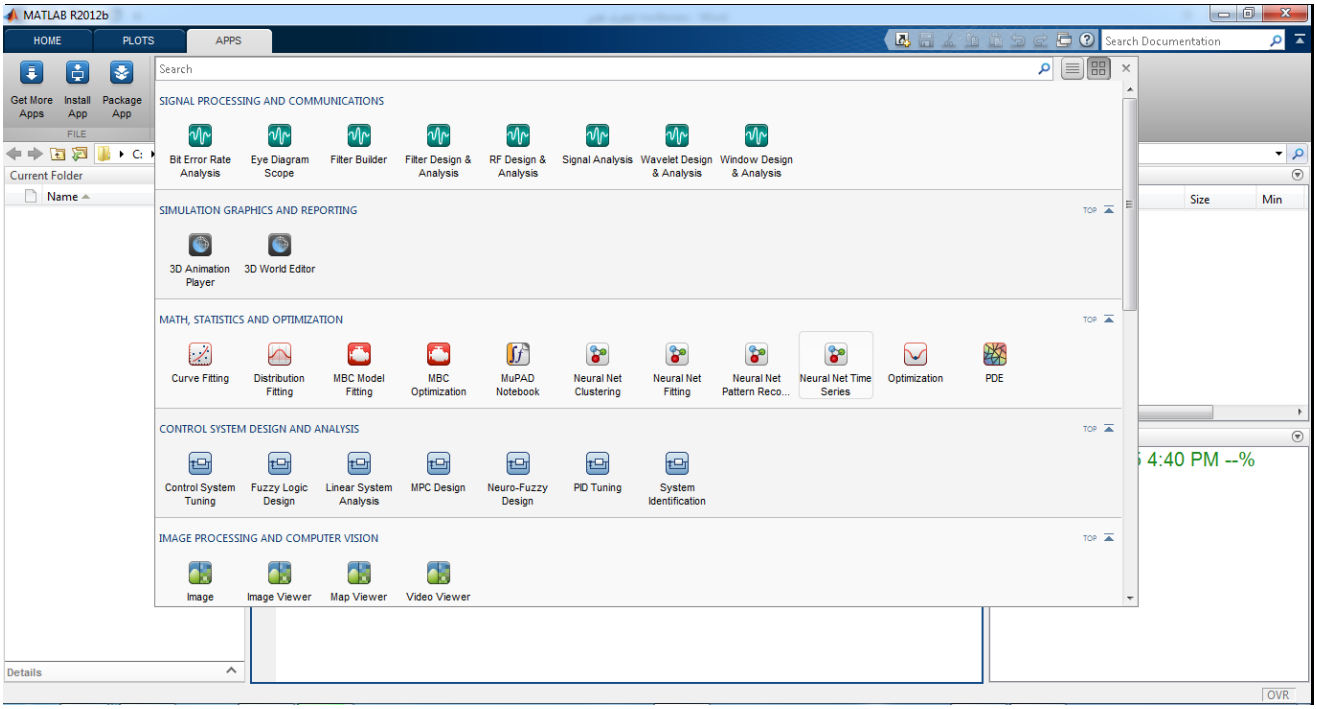
ضمن Toolstrip اضغط على Apps



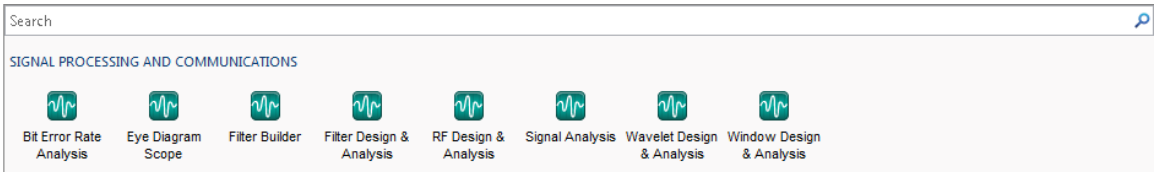
ستجد الواجهة التالية

يمكن رؤية كافة Toolboxes المتاحة لك ضمن نسخة البرمجية، وفتحها عم طريق الضغط على السهم الموضح في الصورة التالية

ستظهر أمامك الواجهة التالية



نلاحظ انها مقسمة وفقاً للاختصاصات أو المجالات التي تدعمها هذه الأدوات. مثلاً توضح هذه الصورة كافة الToolboxes المتخصصة في مجال معالجة الإشارة والاتصالات.



ملاحظة: إن شرح أي أداة toolbox يتطلب مقرراً كاملاً لن نقوم بالخوض في تفاصيل هذه الأدوات ولا بالشرح الهدف هو عرض هذه الغضافات الموجودة ضمن MATLAB، وفي حالة الحاجة لها يمكن التعلم عليها عن طريق MATLAB Help أو عن طريق فيديوهات تعليمية متوفرة على الانترنت أو عن طريق ملفات تزود بها الشركة أو دروس لجامعات مهتمة في الأداة المرغوبة.

الأسئلة

1. اذكر خمسة من توابع التعرف على أنماط المعطيات مع ذكر عملها (Data Type Identification)

مساعدة: قراءة فقرة التعرف على أنماط المعطيات

2. ما هو خرج الرماز التالي:

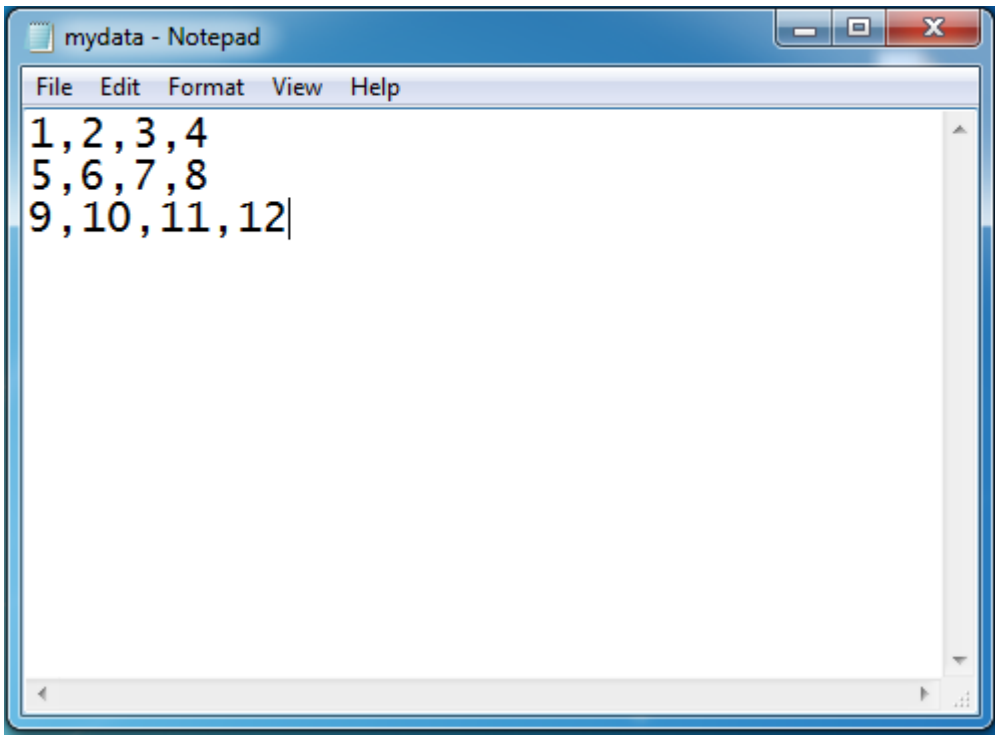
```
x =4;
isinteger(x)
isfloat(x)
isvector(x)
isscalar(x)
isnumeric(x)
```

مساعدة: قراءة فقرة التعرف على أنماط المعطيات

3. اذكر خمسة من توابع التحويل بين أنماط المعطيات مع ذكر عملها (Data Type Conversion)

مساعدة: قراءة فقرة التحويل على أنماط المعطيات

4. في حالة لدينا ملف نصي باسم mydata.txt على الشكل التالي:



ضمن ملف script قم بكتابة رماز برمجي تقوم به أولاً بمسح جميع المتحولات التي تم تعريفها ومسح محتويات Command Window وإغلاق جميع الأشكال ثم باستيراد المعطيات الموجودة ضمنه الخرج هو المعطيات مع معرفة delimiter

مساعدة: قراءة فقرة استيراد المعطيات

5. قم بشرح عمل الرماز التالي بشكل مفصّل:

```
x = 0:0.01:2*pi;
y = [x  ;cos (x)];
fid = fopen('costable.txt', 'w');
fprintf(fid, 't      Cos(t)\n\n');
fprintf(fid, '%f   %f\n', y);
fclose(fid);
```

مساعدة: قراءة فقرة تصدير المعطيات

6. قم بكتابة رماز برمجي من أجل قراءة صورة باسم testch5.png ومن ثم قم بعرضها واستخدم توابع MATLAB لمعرفة بعض المعلومات عن الصورة

مساعدة: قراءة فقرة التعامل مع الصوت

7. قم بشرح الرماز التالي بشكل مفصّل

```
Fs=12e3;
npoints=5*Fs;
sound_rec=wavrecord(npoints,Fs);
wavplay(30*sound_rec,Fs);
```

مساعدة: قراءة فقرة التعامل مع الملفات الصوتية

الإجابات

.1

Function	Purpose
is	Detect state
isa	Determine if input is object of specified class
iscell	Determine whether input is cell array
iscellstr	Determine whether input is cell array of strings
ischar	Determine whether item is character array
isfield	Determine whether input is structure array field
isfloat	Determine if input is floating-point array
ishghandle	True for Handle Graphics object handles
isinteger	Determine if input is integer array
isjava	Determine if input is Java object
islogical	Determine if input is logical array
isnumeric	Determine if input is numeric array
isobject	Determine if input is MATLAB object
isreal	Check if input is real array
isscalar	Determine whether input is scalar
isstr	Determine whether input is character array
isstruct	Determine whether input is structure array
isvector	Determine whether input is vector
class	Determine class of object
validateattributes	Check validity of array
whos	List variables in workspace, with sizes and types

.2

0

1

1

1

1

.3

Function	Purpose
char	Convert to character array (string)
int2str	Convert integer data to string
mat2str	Convert matrix to string
num2str	Convert number to string
str2double	Convert string to double-precision value
str2num	Convert string to number
native2unicode	Convert numeric bytes to Unicode characters
unicode2native	Convert Unicode characters to numeric bytes
base2dec	Convert base N number string to decimal number
bin2dec	Convert binary number string to decimal number
dec2base	Convert decimal to base N number in string
dec2bin	Convert decimal to binary number in string
dec2hex	Convert decimal to hexadecimal number in string
hex2dec	Convert hexadecimal number string to decimal number
hex2num	Convert hexadecimal number string to double-precision number
num2hex	Convert singles and doubles to IEEE hexadecimal strings
cell2mat	Convert cell array to numeric array
cell2struct	Convert cell array to structure array
cellstr	Create cell array of strings from character array
mat2cell	Convert array to cell array with potentially different sized cells
num2cell	Convert array to cell array with consistently sized cells
struct2cell	Convert structure to cell array

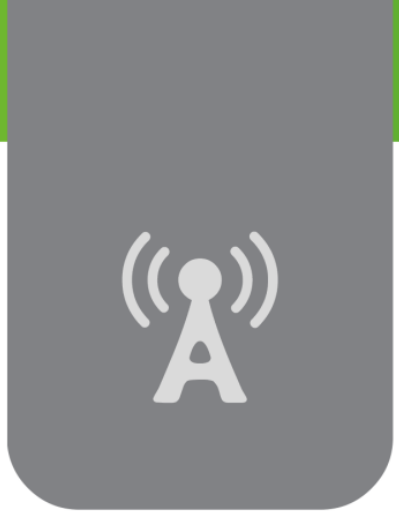
.4

```
%%
 clear all;close all;clc;
 filename = 'mydata.txt';
[A,delimiterOut]=importdata(filename)
```

5. السطر الأول يتم تعريف شعاع من القيم ضمن المجال $[0 - 2\pi]$ بخطوة 0.01
السطر الثاني يتم تعريف مصفوفة بسطرين تحوي على المعطيات التي جرى تعريفها في السطر السابق كشعاع للزمن أم السطر الثاني فهو قيم التابع cos عند النقاط السابقة
السطر الثالث يتم فتح ملف نصي جديد باسم costable بلاحقة txt. وتعريفه ضمن MATLAB على انه متحول يدعى fid
السطر الرابع يتم طباعة ترويسة في داخل الملف وهي t و cos(t) حيث سيتم تخزين القيم ضمن الملف على شكل معطيات ضمن عمودين في سطر نجد قيمة الزمن ومقابلها قيمة التابع cos لهذه النقطة وهو ما يقوم به السطر الخامس
السطر السادس يتم إغلاق الملف.

6.
```
Timage=imread('testch5.png'); % read the image
imshow('testch5.png');% show the image
fo=imfinfo('testch5.png');% get information
```

7. يهدف هذا الرماز لتسجيل مفطع صوتي مدته خمسة ثواني ومن ثم تشغيله
السطر الأول تعريف تردّد التقطيع بقيمة 12 KHz في السطر الثاني تعريف عدد النقاط وهو خمسة أضعاف تردّد التقطيع ومن ثم تسجيل مقطع صوتي باستخدام التعليمة wavrecord يمرر لها عدد النقاط وتردّد التقطيع ومن ثم تشغيل المقطع الصوتي بعد تضخميه 30 مرة.



الفصل السادس: تمارين رياضية باستخدام MATLAB® (جبر خطي، حل جملة معادلات، اشتقاق، تكامل، تحويلات، كثيرات حدود)

Mathematical Exercises using MATLAB® (Linear Algebra, Solving Equations, Derivation, Integration, Transforms and Polynomials)

عنوان الموضوع:

تمارين رياضية باستخدام MATLAB® (جبر خطي، حل جملة معادلات، اشتقاق، تكامل، تحويلات، كثيرات حدود)

Mathematical Exercises using MATLAB® (Linear Algebra, Solving Equations, Derivation, Integration, Transforms and Polynomials)

الكلمات المفتاحية:

MATLAB، الجبر Algebra، الجبر الخطي Linear Algebra، المعادلات الجبرية Algebraic Equations، كثيرات الحدود Polynomials، النهايات Limits، الاشتقاق Derivation، التكامل Integration، المعادلات التفاضلية Differential Equations، التحويلات Transforms، الاحتمالات Probability، الإحصاء Statistics، تحويل لابلاس Laplace Transform، تحويل فورييه Fourier Transform.

ملخص:

تم التركيز في الفصول السابقة على أساسيات اللغة وكيفية التعامل مع البرمجية واستخدمات قدراتها، سنعرض في هذا الفصل عدد من التطبيقات العملية التي يفيد MATLAB في الحصول على نتائج مباشرة لها، سنبدأ بالجبر ونهتم خصوصاً بالجبر الخطي وتطبيقاته وذلك بطريقتين اعتماداً على الحل العدّدي أو بدلالة الرموز، سنعرض حل جملة من المعادلات، والعمليات على المصفوفات. ننتقل بعدها للتعرف على كثيرات الحدود وكيفية التعامل معها باستخدام لغة MATLAB،ومن ثم يتم عرض أكثر المسائل التي يتعرض لها الطالب خلال دراسته أو المهندس خلال عمله وهي قضايا النهايات، الاشتقاق والتكامل حيث سنوضح للمستخدم كيفية حساب المقادير السابقة بشكل يضمن له الدقة والصحة والوقت وذلك عبر عرض عدد من التمارين والأمثلة.
يتم في الجزء اللاحق عرض للمعادلات التفاضلية وحلها ضمن MATLAB، وأخيراً سنعطي لمحة سريعة عن تحويلي لابلاس وفورييه حيث تعتبر هذه التحويلات البنى الأساسية في تحليل الأنظمة أو التعامل مع الإشارت. أخيراً يجري التعرف على تعليمات تفيد في مجال الإحصاء والاحتمالات.

أهداف تعليمية:

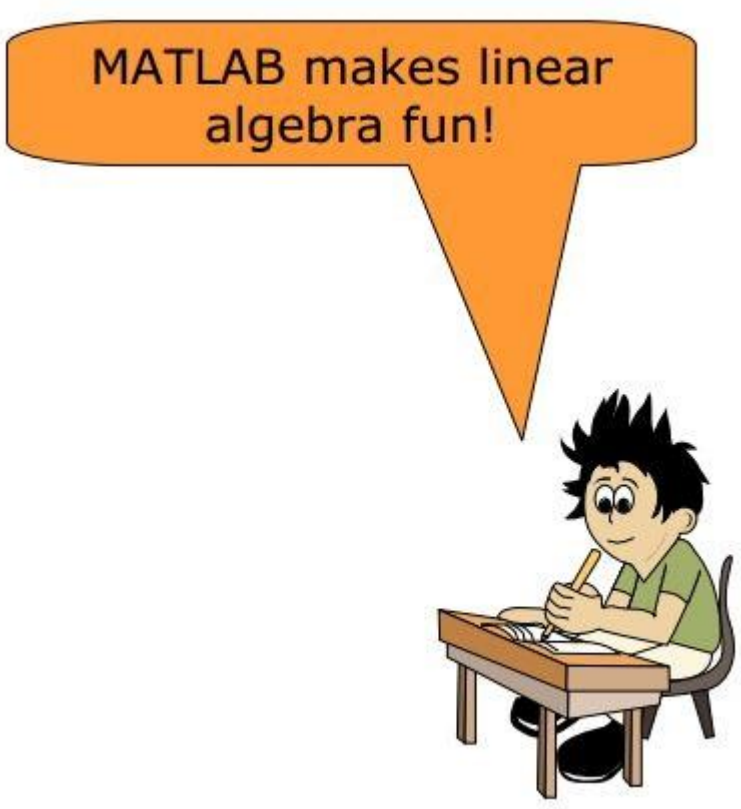
يتعرف الطالب في هذا الفصل على:

- يتقن مفاهيم الجبر، ويستخدم MATLAB للحصول على حلول المعادلات الجبرية.
- يتعرف على كثيرات الحدود وتمثيلها ضمن MATLAB.
- ينجز العديد من المسائل الرياضية كالنهايات والاشتقاق والتكامل باستخدام MATLAB.
- يستخدم MATLAB للحصول على حلول للمعادلات التفاضلية.
- يتعرف على تحويلي لابلاس وفورييه، وبعض المفاهيم في الإحصاء والاحتمالات.

Algebra الجبر

قدمنا في الفصول السابقة عدد كبير من المفاهيم المتعلقة بالبرمجة بشكل عام، وبلغة MATLAB بشكل خاص وقد عرضنا عدد كبير من الميزات التي تقدمها برمجية MATLAB.

نهدف في هذا الفصل إلى تطبيق عدد من المفاهيم الرياضية التي سبق تعلمها سواء في المدرسة أو في دروس السابقة، بهدف توطيد بعض المفاهيم والتعرف على الإمكانيات الهائلة لبرمجية MATLAB التي تسهل عمل المستخدم في معظم الحالات، وفي بعض الأحيان، عندما يصبح إيجاد الحل أو الحصول على علاقة تحليلية معقداً أومستحيلاً ، تكون مثل هذه البرمجيات الطريقة الوحيدة للحصول على الهدف المطلوب.



سنعرض في هذه الفقرة عدة أفكار هي:

- حل المعادلات الجبرية البسيطة من الدرجة الأولى، ثم من الدرجة الثانية ووصولاً إلى درجات أعلى.
- تمثيل جملة معادلات بالشكل المصفوفاتي، وحلها.
- بعض المفاهيم الخاصة بالمصفوفات كالرتبة، والتفريق.
- تجميع وتفريق المعادلات ضمن MATLAB.
- تحليل وتبسيط التعابير الجبرية.

حل المعادلات الجبرية Solving Algebraic Equations

- من أجل تعريف متحول رمزي أو تابع symbolic variable and function يمكن استخدام التعليمة syms و sym.

الأهمية من تعريف مثل هذه المتحولات هو الحصول على الصيغة المجردة للعبارة التحليلية واستخدامها في إيجاد الحلول بدلالة المتحولات ومن ثم القيام بالتعويض العددي عند الحاجة أي أننا نقوم بتجنب الحسابات العدّدية باليد.

نعرض مقارنة بسيطة بين اختيار الحل باستخدام الرموز Symbolics أو الحل العدّدي Numerics.

	المحاسن	المساوئ
Symbolic	الحصول على صيغة تحليلية للحل.	أحياناً لا يمكن الوصول إلى حل.
	معرفة شكل الحل مما قد يفيد في فهم الظاهرة أحياناً.	أحياناً تكون الصيغة النهائية معقدة كثيراً ولا تحمل معنى دون قيمة عدّدية.
Numeric	الحصول على حل ممكن دوماً	من الصعب الحصول على فهم للظاهرة من القيم العدّدية فقط
	ممكن الحصول على حلول دقيقة جداً	**من الممكن أحياناً** الحصول على خطأ في الحل نتيجة خورزميات الحل العدّدي

مثال: نرغب في إيجاد العلاقات التحليلية لمقلوب و محدّد مصفوفة ببعد 2 × 2، قم بكتابة الرماز التالي:

```
%% Mathematical Exercises using MATLAB
clear all; close all; clc;
mat=sym('[a b;c d]');
d=det(mat)
i=inv(mat)
```

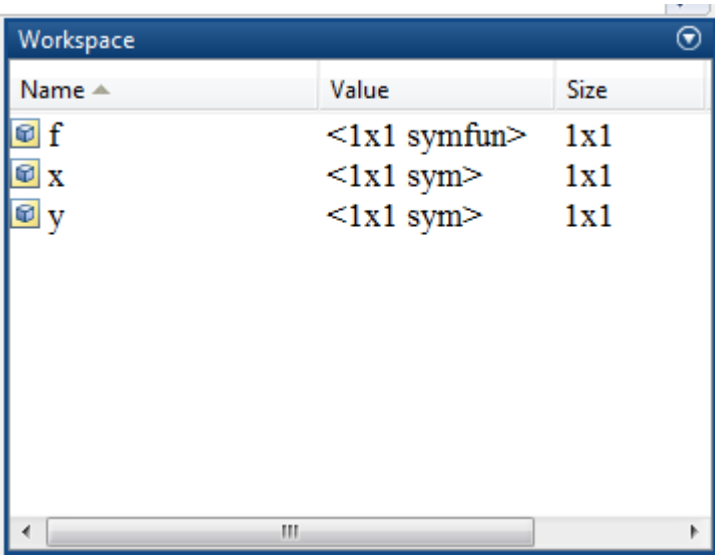
الخرج هو:

```
Command Window
d =

a*d - b*c


i =

[ d/(a*d - b*c), -b/(a*d - b*c)]
[ -c/(a*d - b*c),  a/(a*d - b*c)]

fx >>
```

مثال: لتعريف متحولين x و y، وتابع لمتحولين f(x,y) اكتب الرماز التالي:

```
Editor - C:\Users\Simon\Documents\MATLAB\math_exercises.m
math_exercises.m
1       %% Mathematical Exercises using MATLAB
2       % Algebra
3 -     clear all; close all; clc;
4 -     syms x y % Create symbolic variables x and y using syms.
5 -     syms x y real %  Assume that they are real.
6 -     syms f(x, y) %  Create a symbolic function f that accepts two arguments, x and y.
7 -     f(x, y) = x + 2*y;% Specify the formula for this function.
```

عند التعريف لاحظ المتحولات ضمن فضاء العمل.



أما ضمن Command Window فنلاحظتعريف التابع على الشكل:

```
Command Window
>> f(x, y) = x + 2*y

f(x, y) =

x + 2*y

fx >>
```

ويمكن أيضاً تحديد نمط المتحولين كما في الرماز السابق حيث تم تحديد النمط على أنه حقيقي real.

من أجل رسم التوابع التي تم تعريفها باستخدام الرموز يمكن استخدام التابع ezplot إن المجال الافتراضي الذي يتم الرسم عليه هو $-2\pi < x < 2\pi$ ولكن يجب الانتباه لكون التابع $f(x)$ هو تابع للمتحول x فقط، يوجد خيارات إضافية تسمح لنا بتحديد مجال الرسم أو تحديد رسم تابع لمتحولين يمكن الاطلاع عليها عن طريق help.

مثال:

اكتب الرماز التالي:

```
h = ezplot('x^2')
set(h, 'Color', 'm');
```

الخرج هو رسم للتابع x^2.

ثم اكتب الرماز التالي ولاحظ الخرج:

```
ezplot('x^2-y^4')
```

يقوم التابع برسم حل المعادلة $f(x,y) = 0$.

- من أجل القيام بحل معادلات جبرية يمكن استخدام تعليمة solve.

مثال: قم بكتابة الرماز التالي لحل معادلات بسيطة وتعلم منه كيفية استخدام تعليمة solve، لاحظ وجود طريقتين، إضافةً لإمكانية تحديد المجهول ضمن المسألة.

Editor - C:\Users\Simon\Documents\MATLAB\math_exercises.m

math_exercises.m

```
%%
% Solve Simple Equation
clear all; close all; clc;
% basic form
solve('x - 4')
% assign output to a variable
y = solve('x - 5 =0')
% solve(equation, variable)
solve('v-u-3*t^2=0','v')
% If the right side of an equation is 0,
% you can specify either a symbolic expression
% without the left side or an equation with the == operator.
syms x
solve(x- 1)
solve(x- 2 == 0)
```

الخرج:

```
Command Window
ans =

4


y =

5


ans =

3*t^2 + u


ans =

1


ans =

2

fx >>
```

مثال: حل معادلة من الدرجة الثانية.

```
Editor - C:\Users\Simon\Documents\MATLAB\math_exercises.m
math_exercises.m
23        %% Seconf Order
24 -      clear all; close all; clc;
25 -      syms a b c x
26 -      solve(a*x^2 + b*x + c == 0)
27        %
28 -      eq ='x^2 -7*x + 12 = 0';
29 -      s = solve(eq);
30 -      disp('The first root is: '), disp(s(1));
31 -      disp('The second root is: '), disp(s(2));
```

الخرج على الشكل التالي:

```
Command Window
ans =

 -(b + (b^2 - 4*a*c)^(1/2))/(2*a)
 -(b - (b^2 - 4*a*c)^(1/2))/(2*a)

The first root is:
3

The second root is:
4

fx >>
```

مثال: حل معادلة من درجات أعلى.

Editor - C:\Users\Simon\Documents\MATLAB\math_exercises.m

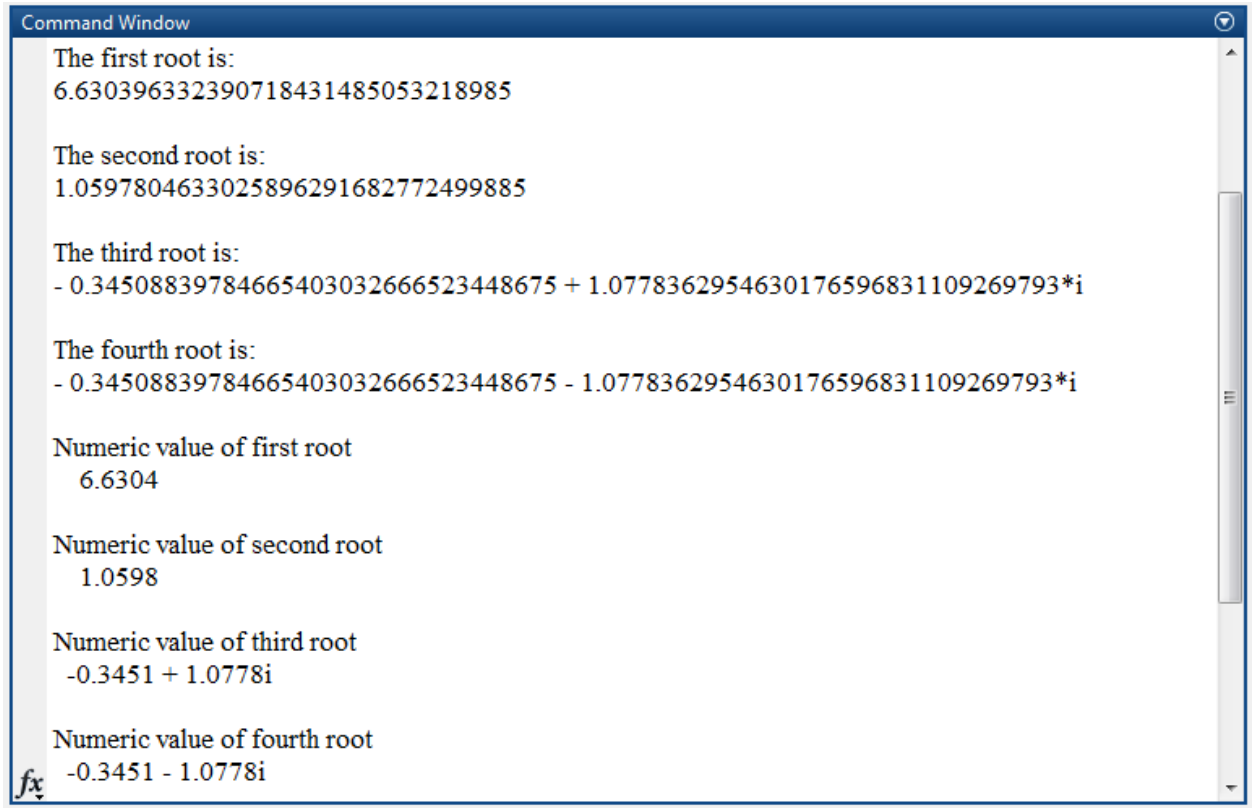
math_exercises.m

```
%% Solving Higher Order Equations
clear all; close all; clc;
% 3rd order
solve('(x-3)^2*(x-7)=0')
% In case of higher order equations, roots are long containing many terms.
% You can get the numerical value of such  roots by converting them to double
eq ='x^4 - 7*x^3 + 3*x^2 - 5*x + 9 = 0';
s = solve(eq);
disp('The first root is: '), disp(s(1));
disp('The second root is: '), disp(s(2));
disp('The third root is: '), disp(s(3));
disp('The fourth root is: '), disp(s(4));
% converting the roots to double type
disp('Numeric value of first root'), disp(double(s(1)));
disp('Numeric value of second root'), disp(double(s(2)));
disp('Numeric value of third root'), disp(double(s(3)));
disp('Numeric value of fourth root'), disp(double(s(4)));
% other options
syms x
solve(x^4 + 1 == 2*x^2 - 1)
```

الخرج:

Command Window

```
ans =

   3
   3
   7
```



حل جملة معادلات باستخدام المصفوفات Solving System of Equations using Matrices

سبق لنا في الفصول السابقة عرض فكرة حل مجموعة من المعادلات عبر تمثيلها بالشكل المصفوفاتي ومن ثم إيجاد الحل كما يلي:

$$A\,x = b$$

$$x = A^{-1}.b$$

وقد وجدنا كيفية إيجاد الحل عبر إيجاد مقلوب مصفوفة، أو باستخدام عملية القسمة، سنستخدم في هذه الفقرة التعليمة solve لحل مجموعة من المعادلات.

مثال:

قم بحل المعادلات التالية:

$$5x + 9y = 5$$

$$3x - 6y = 4$$

قم بكتابة الرماز التالي:

```
%% Solving System of Equations
clear all; close all; clc;
s = solve('5*x + 9*y = 5','3*x - 6*y = 4');
s.x
s.y
```

ولاحظ الخرج، ثم قم بحل جملة المعادلات التالية:

$$x + 3y - 2z = 5$$

$$3x + 5y + 6z = 7$$

$$2x + 4y + 3z = 8$$

المزيد من الجبر الخطي More Linear Algebra

سنتعرف ضمن هذا الرماز على المزيد من التعليمات المفيدة في مجال الجبر الخطي.

مثال:

Editor - C:\Users\Simon\Documents\MATLAB\math_exercises.m

math_exercises.m

```
%% More Linear Algebra
clear all; close all; clc;
% define a MATRIX
mat=[1 2 -3;-3 -1 1;1 -1 1];
%  Calculate the rank of a matrix
r=rank(mat)
 %  Calculate the determinant
d=det(mat)
% Calculate the  matrix inverse
E=inv(mat)
% Eigenvalue decomposition
[V,D]=eig(mat)
```

نفذ الرماز السابق ولاحظ الخرج.

- تعليمة rank تفيد في إيجاد رتبة مصفوفة أي عدد السطور أو الأعمدة المستقلة خطياً وهي تعطي معياراً لوجود مقلوب مصفوفة من عدمه.
- تعليمة det تفيد في إيجاد محدد مصفوفة، لوجود المحّدد يجب أن تكون المصفوفة مربعة، وفي حال كان المحدّد لا يساوي الصفر ومنه يوجد مقلوب للمصفوفة.
- تعليمة inv تفيد في إيجاد مقلوب مصفوفة.

- تعليمة eig تفيد في إيجاد تحليل للمصفوفة السابقة بدلالة القيم الذاتية، يوجد عدد كبير من من طرق تحليل المصفوفات والتي تم برمجتها ضمن MATLAB ومن الممكن استخدامها بسهولة.
- ننصح بالتعرف على البعض التعليمات المفيدة مثلlinsolve ، norm، null، eye، dot، cross، kron.
- يمكن الاستفادة من التعليمات السابقة لبرمجة رمازات فعّالة مثلاً يمكن باستخدام تعليمة det أو rank معرفة مسبقاً أن المصفوفة لها مقلوب أم لا ومن ثم إظهار رسالة خطأ أو تحذير للمستخدم.

تجميع وتفريق المعادلات ضمن MATLAB Expanding and Collecting Equations in MATLAB

قم باستخدام تعليمتي expand و collect.

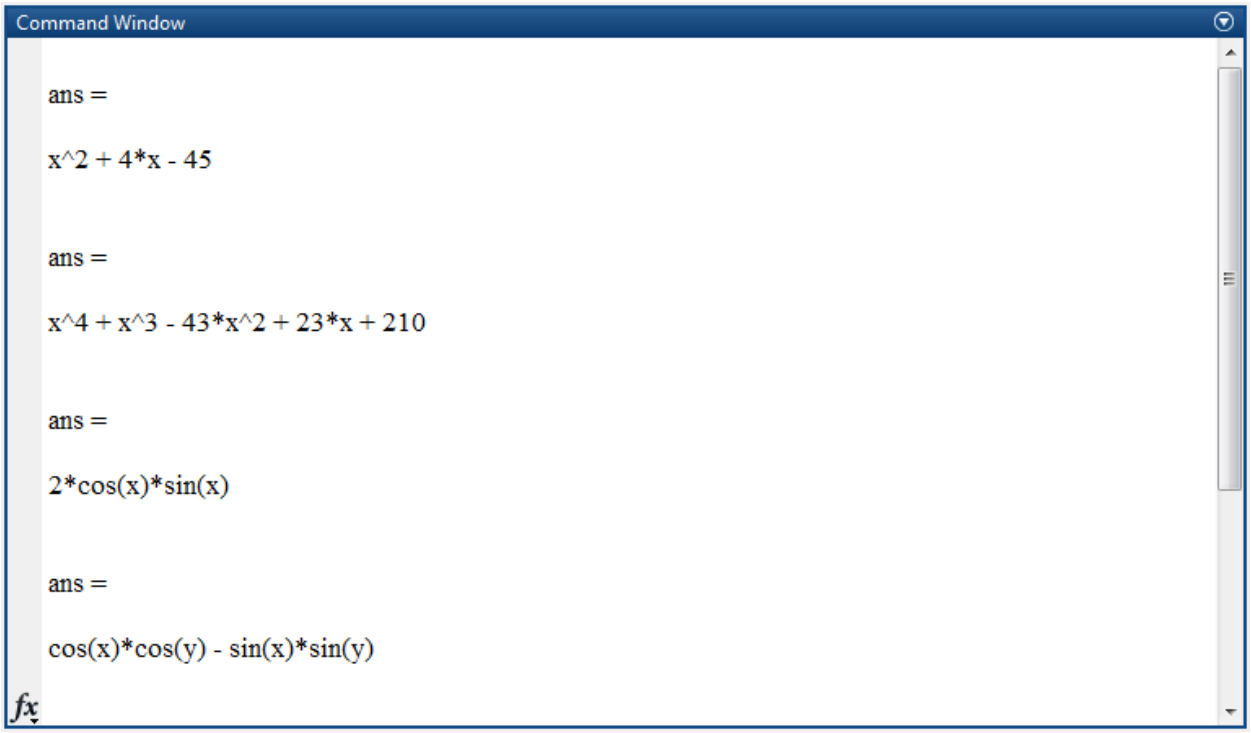
مثال:

Editor - C:\Users\Simon\Documents\MATLAB\math_exercises.m

math_exercises.m

```
83      %%% Expanding and Collecting Equations
84 -    clear all; clc;
85 -    syms x %symbolic variable x
86 -    syms y %symbolic variable x
87      % expanding equations
88 -    expand((x-5)*(x+9))
89 -    expand((x+2)*(x-3)*(x-5)*(x+7))
90 -    expand(sin(2*x))
91 -    expand(cos(x+y))
92      % collecting equations
93 -    collect(x^3*(x-7))
94 -    collect(x^4*(x-3)*(x-5))
```

الخرج:



تحليل وتبسيط التعابير الجبرية

Factorization and Simplification of Algebraic Expressions

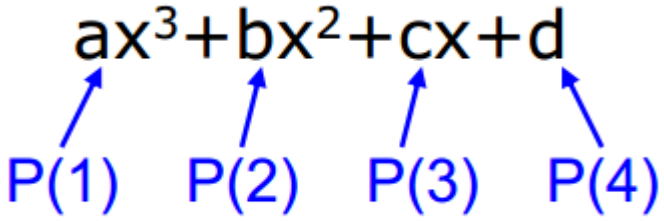
مثال: قم بكتابة الرماز التالي، استخدم تعليمة factor من أجل تحليل العبارات الجبرية وتعليمة simplify من أجل تبسيط التعابير الجبرية، ولاحظ الخرج.

```
%% Factorization and Simplification of Algebraic Expressions
clear all; clc;
syms x %symbolic variable x
syms y %symbolic variable x
factor(x^3- y^3)
factor([x^2-y^2,x^3+y^3])
```

```
simplify((x^4-16)/(x^2-4))
```

كثيرات الحدود Polynomials

نعلم أن معظم التوابع يمكن تمثيلها على شكل كثير حدود برتبة عالية، من أجل تمثيل كثيرات الحدود ضمن MATLAB يكفي توصيف شعاع الأمثال.



لتمثيل كثير الحدود التالي :

$$p(x) = x^4 + 7x^3 - 5x + 9$$

عرف شعاع الأمثال التالي على شكل السطر: p = [1 7 0 −5 9];

$$p(x) = x^3 - 2x - 5$$

عرف شعاع الأمثال التالي على شكل السطر: p = [1 0 −2 −5];

$$p(x) = x^2 - 2$$

عرف شعاع الأمثال التالي على شكل السطر: p = [1 0 −2];

$$p(x) = 2x^3$$

عرف شعاع الأمثال التالي على شكل السطر: p = [2 0 0 0];

أي أننا نحتاج الشعاع p بطول N+1 من أجل تمثيل كثير حدود من الدرجة N.

- من أجل إيجاد جذور كثير حدود استخدام التعليمة roots.

مثال: قم بتجريب الرماز التالي وملاحظة الخرج لكل كثير حدود.

```
%% Mathematical Exercises using MATLAB
% Polynomials
clear all; close all; clc;
p =[1 7 0 -5 9];
r = roots(p);
p1 = [1 -6 -72 -27];
r1 = roots(p1);
p2= [1 -5 2 8];
r2 = roots(p2);
```

- يمكن الحصول على كثير حدود انطلاقاً من جذوره عن طريق التعليمة poly.

مثال: استخدم المتحولات r,r1 ,r2 من المثال السابق واكتب ما يلي: p_e= poly(r) و p_e1= poly(r1) و p_e2= poly(r2) .

- من أجل إيجاد قيمة كثير الحدود عند نقطة يمكن استخدام التعليمة polyval على الشكل y0=polyval(P,x0) حيث x0 هية القيمة المراد إيجاد قيمة كثير الحدود عندها، او يمكن إدخال شعاع من القيم.

مثال:

```
clear all; close all; clc;
p =[1 7 0 -5 9];
polyval(p,4)
p1 = [3 2 1];
polyval(p1,[5 7 9])
```

- يمكن إيجاد أيضاً أفضل كثير حدود يمر من مجموعة نقاط ومن الدرجة التي يرغب المستخدم بها عن طريق التعليمة polyfit.

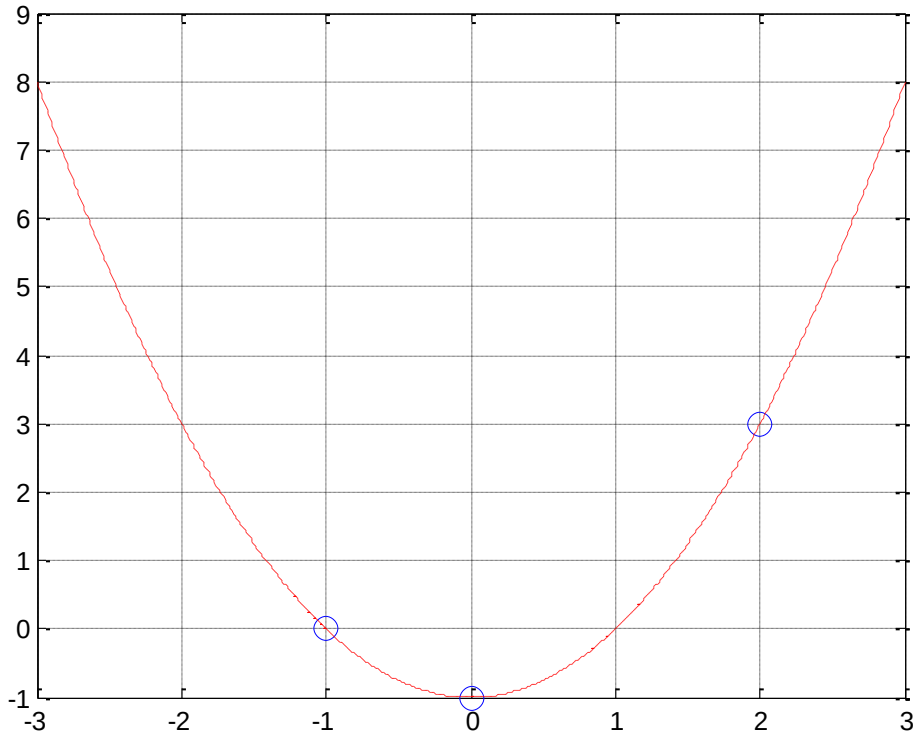
مثال:

Editor - C:\Users\Simon\Documents\MATLAB\math_ex2.m

math_ex2.m

```
17      %% Polynomial fitting
18 -    clear all; close all; clc;
19      % data vectors
20 -    X=[-1 0 2];
21 -    Y=[0 -1 3];
22 -    p2=polyfit(X,Y,2);
23      % finds the best second order polynomial that fits the points
24      % (-1,0),(0,-1), and (2,3)
25 -    plot(X,Y,'o', 'MarkerSize', 10); hold on; grid on;
26 -    x = -3:.01:3;
27 -    plot(x,polyval(p2,x),'r--');
```

الخرج:



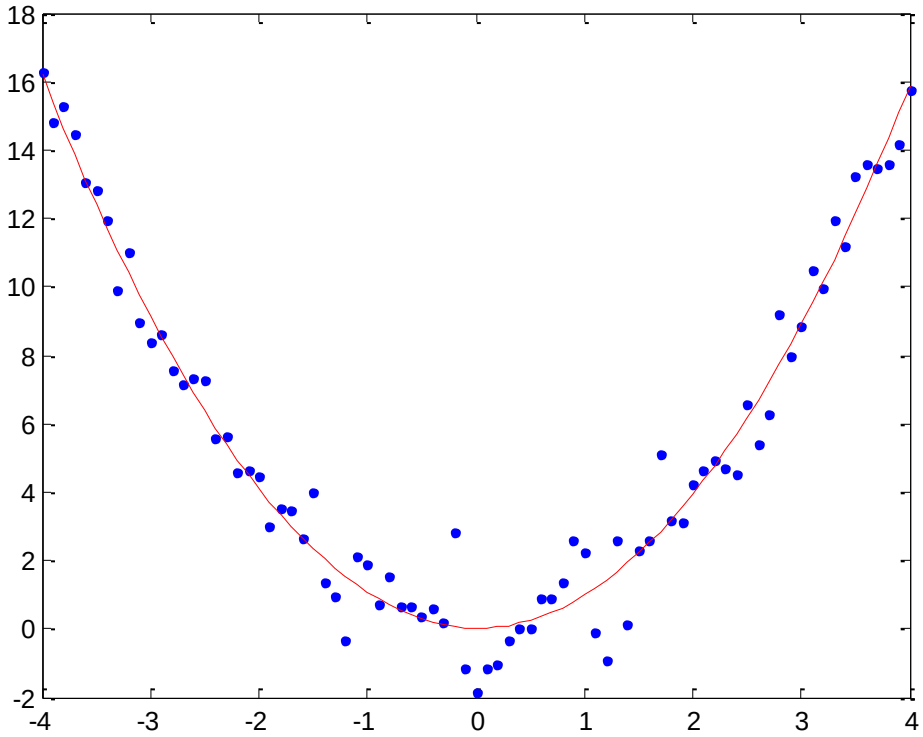
مثال: سنقوم باستخدام التعليمات السابقة على التابع $f(x) = x^2$ ولكن سنضيف هذه المرة إلى التابع ضجيج عشوائي.

Editor - C:\Users\Simon\Documents\MATLAB\math_ex2.m

math_ex2.m

```
28      %%
29 -    clear all; close all; clc;
30      % Evaluate y = x^2 for x=-4:0.1:4.
31 -    x=-4:0.1:4;
32 -    y=x.^2;
33      % Add random noise to these samples. Use randn.
34 -    y=y+randn(size(y));
35      % Plot the noisy signal with .markers
36 -    plot(x,y,'.');
37      % Fit a 2nd degree polynomial to the noisy data
38 -    p=polyfit(x,y,2);
39      % Plot the fitted polynomial on the same plot
40      % using the same x values and a red line
41 -    hold on;
42 -    plot(x,polyval(p,x),'r')
```

الخرج:



النهايات Limits

سنتعرف في هذه الفقرة على طريقة حساب النهايات ضمن MATLAB، سنعرض بعض التمارين ونتأكد من بعض خواص النهايات. يتم استخدام التابع limit، يأخذ التابع الأشكال القواعدية التالية:

Syntax

```
limit(expr,x,a)
limit(expr,a)
limit(expr)
limit(expr,x,a,'left')
limit(expr,x,a,'right')
```

إن التعليمة السابقة تأخذ على دخلها تعبير expression وتقوم بحساب النهاية له عند لبصفر في حالة عدم تحديد قيمة النهاية المرجوة.

مثال:

Editor - C:\Users\Simon\Documents\MATLAB\math_limits.m

math_limits.m

```
1     %% Mathematical Exercises using MATLAB
2     % Limits
3 -   clear all; close all; clc;
4 -   syms x
5 -   limit((x^3+3)/(x^4+4))
```

النتيجة هي 3/4.

مثال: لحساب النهاية عند قيمة معينة، بعد تعريف المتحول x syms اكتب التعليمة التالية:

```
limit((x -3)/(x-1),1)
```

النهاية هي NAN.

لناخذ مثال آخر،

```
limit(x^2+1,3)
```

الجواب هو 10.

مثال: سنقوم ضمن هذه المثال بالتاكد من خواص النهايات طبعاً مع احترام جميع الشروط الرياضية اللازمة لوجود النهاية، نكتب:

$$\lim_{x\to n}(f(x)+g(x)) = \lim_{x\to n} f(x) + \lim_{x\to n} g(x)$$

$$\lim_{x\to n}(f(x)-g(x)) = \lim_{x\to n} f(x) - \lim_{x\to n} g(x)$$

$$\lim_{x\to n}(f(x).g(x)) = \lim_{x\to n} f(x).\lim_{x\to n} g(x)$$

$$\lim_{x\to n}(f(x)/g(x)) = \lim_{x\to n} f(x) / \lim_{x\to n} g(x)$$

لنقم بتعريف تابعين f و g، ومن ثم نقوم بحساب نهاية كل من التابعين عندما يسعى المتحول x إلى القيمة 4، ومن ثم سنقوم بالتأكد من خواص النهايات السابقة باستخدام MATLAB. حيث نعرّف:

$$f(x) = \frac{3x+5}{x-3}$$

$$g(x) = x^2 + 1$$

قم بإنشاء script واكتب الرماز التالي:

Editor - C:\Users\Simon\Documents\MATLAB\math_limits.m

math_limits.m

```
%% Mathematical Exercises using MATLAB
% Limits
clear all; close all; clc;
syms x
f =(3*x +5)/(x-3);
g = x^2+1;
L1 = limit(f,4)
L2 = limit (g,4)
LAdd = limit(f + g,4)
LSub = limit(f - g,4)
LMult = limit(f*g,4)
LDiv = limit (f/g,4)
```

الخرج هو:

مثال: سنناقش في هذا المثال حالة وجود انقطاع في التابع ما وليكن $f(x)$ عند بعض قيم المتحول x، حيث تكون النهاية غير موجودة عند نقاط الانقطاع تلك، أي تكون النهاية من اليمين (من أجل قيم للمتحول أكبر من قيمة نقطة انقطاع) لا تساوي النهاية من اليسار(من أجل قيم للمتحول أصغر من قيمة نقطة انقطاع).

لنعرف التابع التالي:

$$f(x) = \frac{x-3}{|x-3|}$$

سنبين ضمن التمرين أن $\lim_{x\to 3} f(x)$ غير موجودة وذلك بطريقتين:

- رسم منحني التابع وإظهار نقاط الانقطاع.
- حساب النهايات وإظهار أن النهايتين عند 3 ← x (من اليمني ثم من اليسار) غير متساويتين.

يتم حساب النهاية من اليمين right، أو اليسار left عبر تمرير كلمة 'left' أو 'right' ضمن متحولات التعليمة.

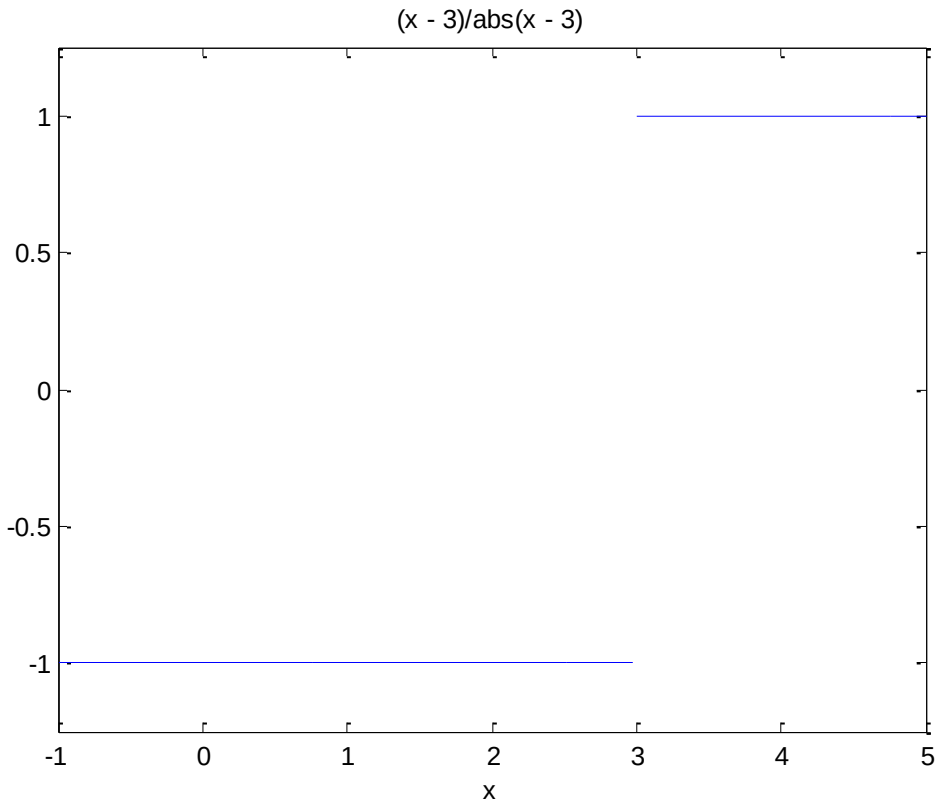
لنكتب الرماز التالي:

```
%%
```

```
clear all; close all; clc;
syms x
f =(x -3)/abs(x-3);
ezplot(f,[-1,5])
L = limit(f,x,3,'left')
R = limit(f,x,3,'right')
```

الخرج هو:

```
Command Window
L =
-1
R =
1
fx >>
```



الاشتقاق Derivation

يتم حساب المشتقات ضمن MATLAB عن طريق التعليمة diff وذلك من أجل التوابع التي جرى تعريفها باستخدام متحولات Symbolic.

في الشكل البسيط للتعليمة يمكن تمرير التابع الذي نرغب بإيجاد المشتق له للتابع، لنقم باستخدام التعليمة diff لإيجاد مشتق التابع التالي:

$$f(x) = 3x^2 + 2x^{-2}$$

قم بإنشاء script واكتب الرماز التالي ولاحظ الخرج.

```
%% Mathematical Exercises using MATLAB

% Derivation

clear all; close all; clc;

syms x

f =3*x^2+2*x^(-2);

diff(f)
```

يمكن كمثال التأكد من خواص الاشتقاق (الخطية، مشتق جمع وطرح تابعين، مشتق جداء ، ...) أو من الممكن أيضاً إيجاد المشتقات لبعض التوابع الأساسية، نعرض في الجدول التالي بعض التوابع الأساسية مع مشتقاتها ومن ثم سنقوم بإيجادها باستخدام MATLAB.

Function	Derivative
$c^{a.x}$	$c^{a.x}$.ln c.a (ln is natural logarithm)
e^x	e^x
ln x	1/x
$\ln_c x$	1/x.ln c
x^x	x^x.(1 + ln x)
sin(x)	cos(x)
cos(x)	-sin(x)
tan(x)	$\sec^2(x)$, or $1/\cos^2(x)$, or $1 + \tan^2(x)$
cot(x)	$-\csc^2(x)$, or $-1/\sin^2(x)$, or $-(1 + \cot^2(x))$
sec(x)	sec(x).tan(x)
csc(x)	-csc(x).cot(x)

مثال: سنقوم بإيجاد المشتقات لبعض التوابع. اكتب الرماز التالي ولاحظ الخرج وقم بالحسابات النظرية يدوياً أيضاً ومن ثم قم بالمقارنة.

Editor - C:\Users\Simon\Documents\MATLAB\math_der.m

math_der.m

```
%%%
clear all; close all; clc;
syms x
y = exp(x); diff(y)
y1 = x^9; diff(y1)
y2 = sin(x); diff(y2)
y3 = tan(x); diff(y3)
y4 = cos(x); diff(y4)
y5 = log(x); diff(y5)
y6 = log10(x); diff(y6)
y7 = sin(x)^2; diff(y7)
y8 = cos(3*x^2+2*x +1); diff(y8)
y9 = exp(x)/sin(x); diff(y9)
```

مثال: سنقوم ضمن هذا المثال بحساب مشتقات من مراتب عليا لتابع ما.

لحساب مشتقات تابع f من مراتب أعلى يمكن استخدام تعليمة diff لكن مع إضافة مرتبة المشتق باستخدام الشكل القواعدي التالي: diff(f,n). مثلاً لحساب المشتق الثاني للتابع $f(x) = x.e^{-3x}$ نكتب الرماز التالي:

```
syms x

f = x*exp(-3*x);

diff(f,2)
```

سنقوم بتنفيذ مثال آخر على استخدام التابع diff، لكن هذا المرة مع استخدام تعليمات أكثر حيث نرغب في معرفة تحقق معادلة من عدمها.

نعرّف التابع $f(x) = 3\sin(x) + 7\cos(5x)$ ونرغب في معرفة فيما إذا كانت المعادلة التالية محققة:

$$f'' + f = -5\cos(2x)$$

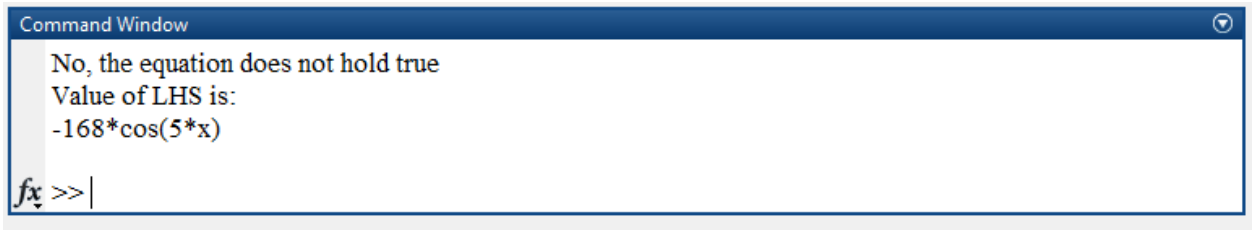
نكتب الرماز التالي:

Editor - C:\Users\Simon\Documents\MATLAB\math_der.m

math_der.m

```
%%%
clear all; close all; clc;
syms x
y =3*sin(x)+7*cos(5*x);% defining the function
lhs = diff(y,2)+y;%evaluting the lhs of the equation
rhs =-5*cos(2*x);%rhs of the equation
if(isequal(lhs,rhs))
disp('Yes, the equation holds true');
else
disp('No, the equation does not hold true');
end
disp('Value of LHS is: '), disp(lhs);
```

الخرج:



مثال: سنقوم ضمن هذا المثال بإيجاد القيمة العظمى والدنيا لتابع ما.

البحث عن قيمة عظمى محلية أو صغرى لتابع ما $y = f(x)$، هذا يعني أننا نبحث عن النقاط التي تحقق المعادلة التالية:

$$f'(x) = 0$$

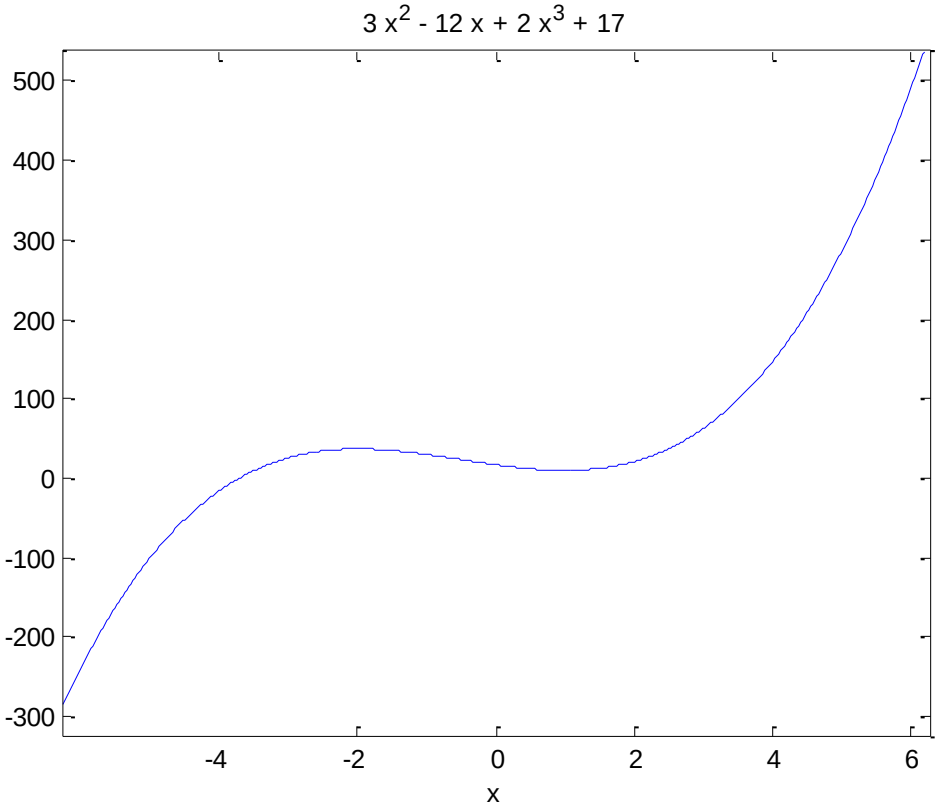
لنطبق ما سبق على التابع التالي:

$$f(x) = 2x^3 + 3x^2 - 12x + 17$$

سنقسم العمل ضمن المثال التالي إلى عدة خطوات:

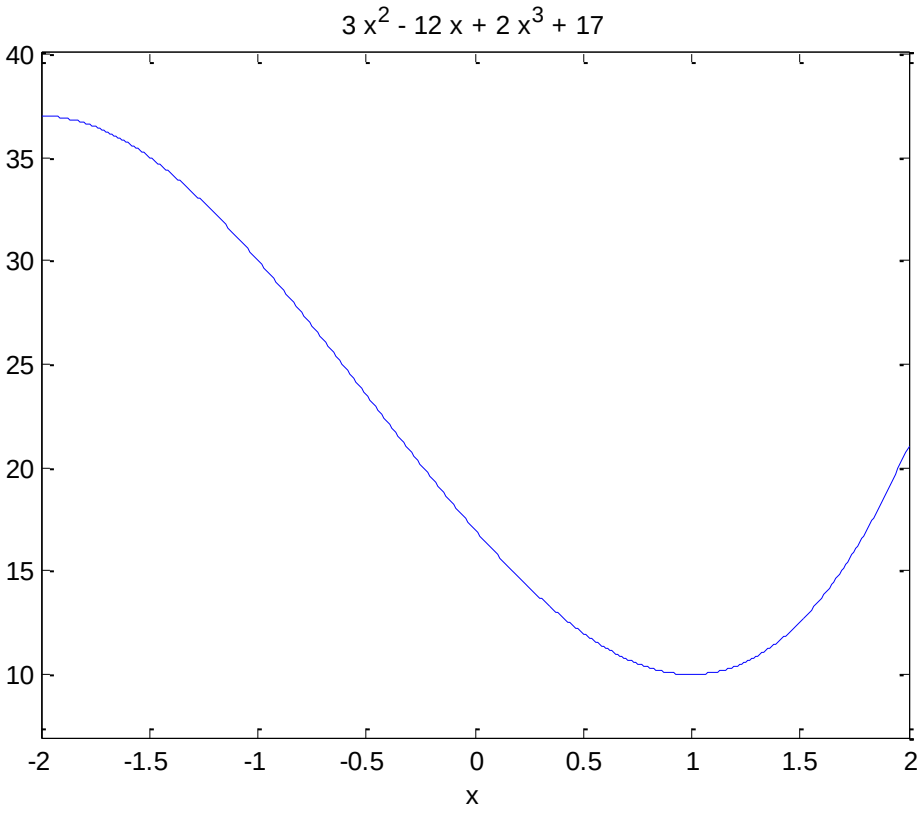
1. تعريف التابع ورسمه:

```
%%
clear all;clc;close all;
syms x
y =2*x^3+3*x^2-12*x +17;% defining the function
ezplot(y)
```



2. الهدف هو إيجاد القيمة المحلية الصغرى والعظمى على المجال [2,2−] لذلك سنقوم برسم التابع على المجال السابق.

```
ezplot(y,[-2,2])
```



3. ثم نقوم بإيجاد المشتق:

```
g = diff(y)
```

نحصل على الجواب التالي:

```
g=  6*x^2 + 6*x - 12
```

4. الآن نقوم بحل المعدلة g=0.

```
s = solve(g)
```

نحصل على الجواب التالي:

```
s = 1  ,  -2
```

5. تتفق النتيجة السابقة مع رسم التابع، بعد ذلك نقوم بحساب قيمة التابع f عند النقاط x=1, −2 ويمكن ذلك باستخدام التعليمة subs.

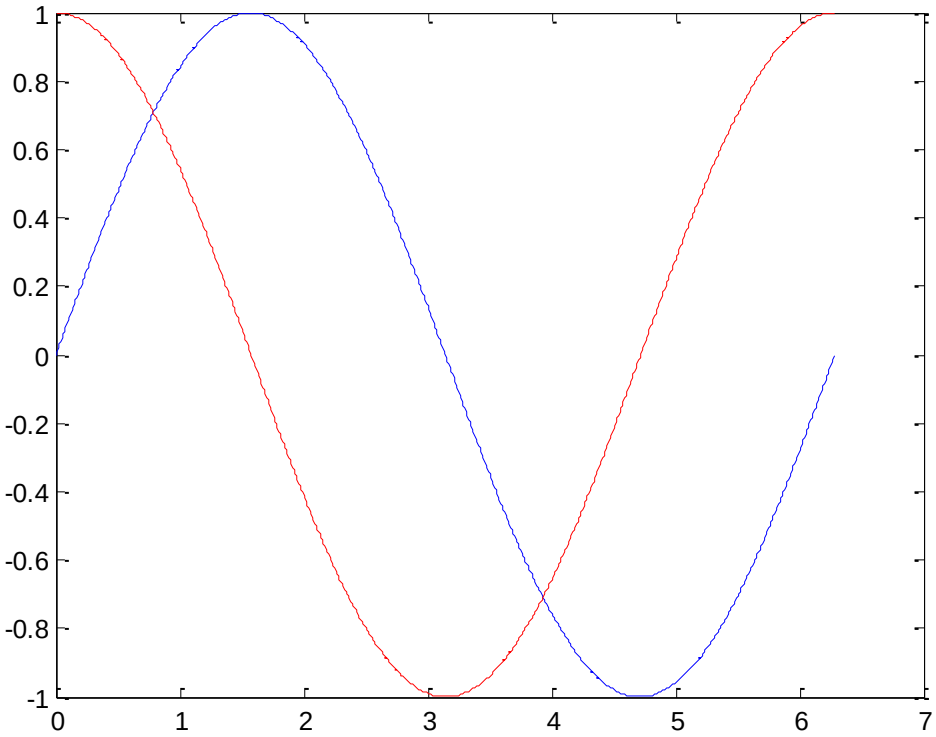
```
subs(y,1), subs(y,-2)
```

نحصل على الجواب التالي:

```
ans = 10
ans = 37
```

مثال: يمكن تطبيق تعليمة diff مع أمثلة عددية كما في المثال التالي:

```
x=0:0.01:2*pi;
y=sin(x);
dydx=diff(y)./diff(x);
plot(x,y,x(2:end),dydx,'r')
```



يجب الانتباه إلى أن الشعاع الناتج عن استخدام تعليمة diff يكون بعدد نقاط أقل من شعاع الدخل للتابع بنقطة واحدة وهذا طبيعي كونه يأخذ الفرق بين نقطتين لإيجاد تقريب للمشتق.

التكامل Integration

تفيد حساب التكاملات في عدد كبير من المسائل مثل إيجاد مساحة أو حجم جسم ما، عمل قوة، مركز الثقل لجسم ما، استطاعة إشارة ما إضافةً لتطبيقات عدديّة مختلفة وواسعة. نقدم في هذه الفقرة فكرة عن إمكانيات MATLAB في حساب التكاملات بنوعيها:

- النوع الأول هو التكامل غير المحدود والذي يفيد في إيجاد التابع الأصلي أي ما هو معاكس لعملية الاشتقاق.
- النوع الثاني هو التكامل المحدود أي إيجاد قيمة تكامل التابع على مجال ما.

1. إيجاد التكامل غير المحدود (التابع الأصلي) Finding Indefinite Integral:

بالتعريف، إذا كان مشتق التابع $f(x)$ هو $f'(x)$، عندها يمكننا القول أن التكامل غير المحدود (التابع الأصلي) لـ$f'(x)$ بالنسبة لـx هو التابع $f(x)$.

مثلاً: نعلم ان المشتق بالنسبة ل x للتابع $f(x) = x^2$ هو $f'(x) = 2x$ عندها يكون التكامل غير المحدود (التابع الأصلي) للتابع $2x$ هو x^2. باستخدام الرموز:

$$f'(x) = 2x \Rightarrow \int 2x\,dx = x^2$$

يجب التنبيه أن التكامل غير المحدود ليس وحيداً، حيث أن مشتق التابع $x^2 + c$ ، حيث c ثابت اختياري، هو أيضاً التابع $2x$. أي نكتب:

$$\int 2x\,dx = x^2 + c$$

حيث c ثابت اختياري.

يتيح MATLAB لمستخدميه تعليمة int وذلك لحساب تكامل تعبير expression ما. للحصول على التكامل غير المحدود نكتب int(f).

مثلاً:

```
%% Mathematical Exercises using MATLAB
% Integration
clear all; close all; clc;
syms x
int(2*x)
```

الخرج هو ans = x^2.

مثال: سنقوم في هذا المثال في إيجاد التابع الأصلي لعدد من التوابع المستخدمة كثيراً، قم بإنشاء ملف script من أجل كتابة الرماز التالي:

```
%%
syms x n
int(sym(x^n))
f ='sin(n*t)';
int(sym(f))
syms a t
int(a*cos(pi*t))
int(a^x)
```

الخرج

```
Command Window
ans =

piecewise([n == -1, log(x)], [n ~= -1, x^(n + 1)/(n + 1)])


ans =

-cos(n*t)/n


ans =

(a*sin(pi*t))/pi


ans =

a^x/log(a)

fx >>
```

مثال: سنقوم أيضاً في حساب بعض التكاملات لتوابع شهيرة ومستخدمة كثيراً لكن الإضافة في هذا التمرين هو استخدام تعليمة pretty التي تساعد في جعل النتيجة أسهل للقراءة على المستخدم، قم بتنفيذ الرماز التالي ولاحظ الخرج.

```
Editor - C:\Users\Simon\Documents\MATLAB\math_int.m
math_int.m
14        %%
15 -      clear all; close all; clc;
16 -      syms x n
17 -      int(cos(x))
18 -      int(exp(x))
19 -      int(log(x))
20 -      int(x^-1)
21 -      int(x^5*cos(5*x))
22 -      pretty(int(x^5*cos(5*x)))
23 -      int(x^-5)
24 -      int(sec(x)^2)
25 -      pretty(int(1-10*x +9* x^2))
26 -      int((3+5*x -6*x^2-7*x^3)/2*x^2)
27 -      pretty(int((3+5*x -6*x^2-7*x^3)/2*x^2))
```

2. إيجاد التكامل المحدود Finding Definite Integral:

نحصل على قيمة التكامل المحدود على مجال [a,b] عن طريق العلاقة التالية:

$$\int_a^b f(x)\,dx = F(b) - F(a)$$

أي إيجاد قيمة التابع الأصلي عند أطراف المجال، يستفاد من التكامل المحدود في حساب مساحة تابع بين منحني التابع ومحور السينات x-axis. من أجل حساب هذا التكامل ضمن MATLAB يمكن استخدام تعليمة int السابقة ولكن يجب تحديد مجال التكامل أي نكتب int(f,a,b).

مثلاً لحساب التكامل التالي $\int_4^9 x\,dx$ نكتب int(x,4,9) وتكون النتيجة هي $\frac{65}{2}$.

مثال: سنقوم ضمن هذا المثال بحساب التكامل بين محور السينات ومنحني التابع $f(x) = x^3 - 2x^2 + 5$ ضمن المجال [1,2]. اكتب الرماز التالي ضمن ملف script، ولاحظ قيمة الخرج حيث من الممكن القيام بحساب القيمة يدوياً للمقارنة.

```
%% Definite Integral
clear all; close all; clc;
syms x
f = x^3-2*x +5;
a =int(f,1,2);
display('Area: '), disp(double(a));
```

مثال: حساب تكامل محدود يتطلب تكامل بالتجزئة واستخدام التعليمة subs لتعويض قيم طرفي المجال.

إن حساب التكامل بالتجزئة يتطلب عدداً إضافياً من الحسابات عادةً يمكن استخدام MATLAB لإجراء هذه الحسابات، نرغب في حساب التكامل:

$$\int_a^b xe^x\,dx$$

ومن ثم حساب هذه القيمة ضمن المجال [a=0,b=2].

```
%%
clear all; close all; clc;
syms a b x
Q=int(x*exp(x),a,b)
subs(Q,{a,b},{0,2})
```

ملاحظة: يحتوي MATLAB أيضاً على طرائق عدّية أو تقريبات لحساب التكاملات نذكر منها Adaptive Simpson's quadrature و Trapezoidal، ويوجد توابع منجزة ضمن MATLAB للقيام بالحسابات، نذكر كمثال على الطريقتين السابقتين:

```
q=quad(@(x) sin(x),0,pi);
x=0:0.01:pi;
```

```
z=trapz(x,sin(x));
```

المعادلات التفاضلية Differential Equations

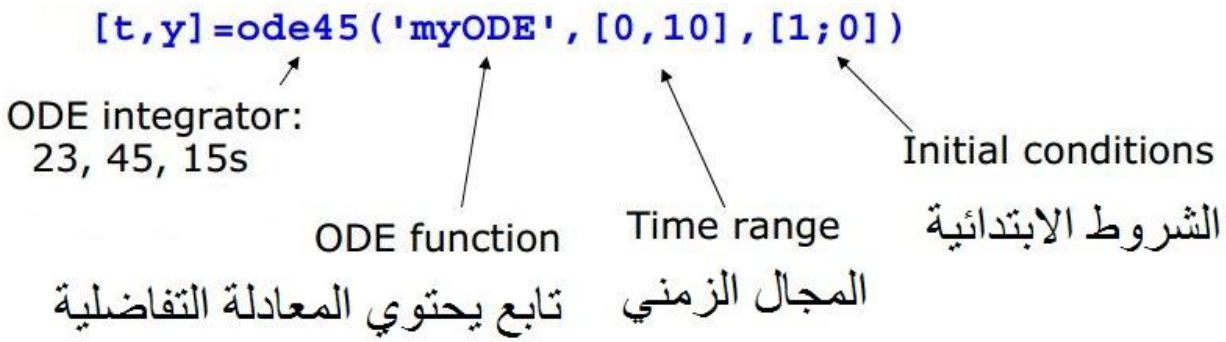
تمثل المعدلات التفاضلية عنصراً أساسياً في غالبية العلاقات الموصفة للحركة جسم ما، كما أنها مضمَّنة في العديد من الظواهر الفيزيائية والتطبيقات العملية، سنعرض في هذه الفقرة كيفية الحصول على حلول للمعدلات التفاضلية باستخدام MATLAB.

نعلم مسبقاً أن حل المعادلات التفاضلية يتم عن طريق التكامل عدد من المرات وفقاً لمرتبة المعادلة التفاضلية، عادةً يتم تنجيز حلول المعادلات التفاضلية باستخدام خوارزميات التحليل العدّدي يتم ضمن MATLAB تجميع هذه الخوارزميات وفقاً للطريقة التي تعتمد عليها في الحل، والمشكلات التي تسطيع حلها بما يعرف بـ Solver، ويشتمل MATLAB على عدد من solvers الخاصة بحلول المعادلة التفاضلية الاعتيادية Ordinary Differential Equation ODE.

يضمن الاختيار الصحيح لـ ODE Solver الحفاظ على كثير من وقت المستخدم، إضافةً للحصول على نتائج دقيقة، نعرض بعض من ODE Solvers (هذه الأسماء هي نفس اسم التعليمة ضمن MATLAB):

1. ode23 وهو Low-order solver، يتم استخدامه عندما تجري عملية المكاملة على مجال صغير او عندما تكون الدقة أقل أهميةً من السرعة.
2. ode45 وهو High-order Solver، يعتبر أكثر استخداماً من باقي الـsolvers، يتميز بدقة عالية وسرعة مقبولة.
3. ode15s يستخدم عندما تحتوي المعادلة التفاضلية على ثوابت زمنية تختلق تبعاً لمرتبة المتحولات ضمن المعادلة ومطالاتها.

من أجل استخدام التعليمات لدينا الشكل القواعدي التالي:



الدخل:

1. اسم التابع الذي يحتوي على المعادلة التفاضلية الاعتيادية ODE، أو استخدام تابع من نمط anonymous، يحتوي التابع على دخلين مثلاً (t,y) ويعيد القيمة dy/dt. ويجب الانتباه أن جملة المعادلة التفاضلية يجب أن تكون من المرتبة الأولى حتى نستطيع استخدام ODE Solvers وحتى لو كانت غير خطية.

2. المجال الزمني: هو عبارة عن شعاع يحتوي على قيمتين تحدّدان بداية ونهاية الزمن للمحاكاة.
3. الشروط الابتدائية: شعاع عمود يحتوي على الشروط الابتدائية من أجل كل معادلة تفاضلية اعتيادية ODE.

الخرج:

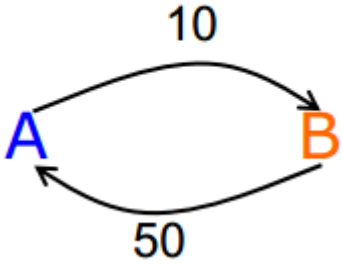
1. t يحتوي على النقاط التي تمثل الزمن.
2. y يحتوي على قيم الحل الموافق للمجال الزمني، حيث ان كل سطر ضمن y هو موافق لقيم المجال الزمني ضمن سطر ضمن المتحول t.

مثال: سيتم ضمن هذا المثال شرح آلية سير تفاعل كيميائي، أولاً يجب بناء تابع يوصف الحالة هو ODE function والذي يجب أن يعيد قيم المشتق من أجل لحظات زمنية معينة وقيم دخل معينة.

المعادلات التي توصف التفاعل الكيميائي هي:

$$\frac{dA}{dt} = -10A + 50B$$

$$\frac{dB}{dt} = 10A - 50B$$



نقوم أولاً بإنشاء ملف تابع function يحتوي على الرماز التالي، لا تنسى أن يكون التابع ضمن نفس مسار العمل عند استعماله ولا تنسى أن يكون اسم الملف المحفوظ به التابع هو نفس اسم التابع.

```
function dydt = chem (t,y)
% this function describe Chemical Reaction ODE function
dydt = zeros(2,1);
dydt(1)=-10*y(1)+50*y(2);
dydt(2)=10*y(1)-50*y(2);
end
```

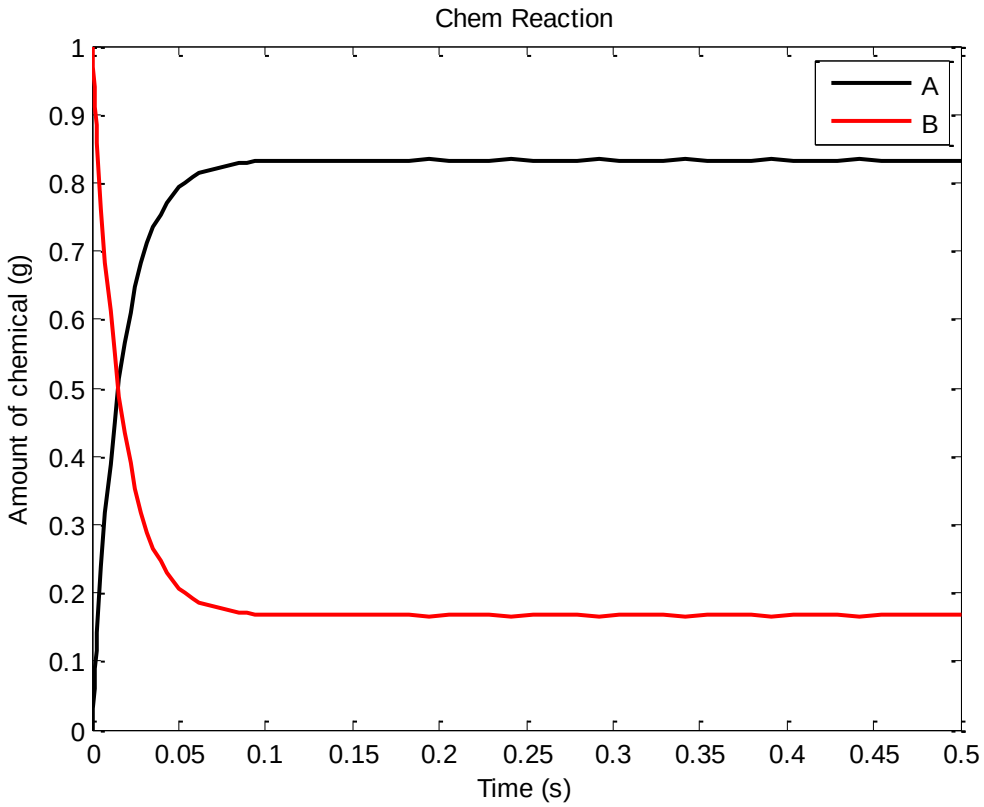
قم بإنشاء ملف script واكتبه ضمنه الرماز التالي:

Editor - C:\Users\Simon\Documents\MATLAB\ode_exp.m

ode_exp.m

```
1   %% ODE
2   % assumes that only chemical B exists initially
3 - [t,y]=ode45('chem', [0 0.5] , [0 1] );
4 - plot(t,y(:,1),'k','LineWidth',1.5);
5 - hold on;
6 - plot(t,y(:,2),'r','LineWidth',1.5);
7 - legend('A','B');
8 - xlabel('Time (s)');
9 - ylabel('Amount of chemical (g)');
10 - title('Chemical Reaction');
```

الخرج هو:



ملاحظة: يستخدم MATLAB's ODE Solvers خطوة زمنية متغيرة، في بعض الحالات من المفيد أخذ خطوة زمنية ثابتة.

من الممكن القيام بذلك عن طريق تمرير شعاع زمن محدّدة الخطوة ضمنه، أما الخرج فيكون عند اللحظات التي جرى تحديدها ضمن شعاع الزمن، إن تحديد الخطوة الزمنية ضمن شعاع الدخل يؤدي إلى زيادة زمن تنفيذ التعليمة وذلك لأن قيم التابع يتم إجراء استيفاء interpolation لإعطاء الخرج عند اللحظات المحدّدة.

مثلاً جرّب ذلك على المثال السابق:

```
[t1,y1]=ode45('chem',0:0.001:0.5,[0 1]);
```

يمكن استخدام toc tic للتحقق أيضاً.

يمكن أيضاً تحديد خيارات إضافية للحل منها تحديد التسامح في الخطأ عن طريق التعليمة odeset.

مثال:

```
options=odeset('RelTol',1e-6,'AbsTol',1e-10);
[t,y]=ode45('chem',[0 0.5],[0 1],options);
```

لمعرفة أكثر عن هذه الخيارات استخدم help.

مثال: استخدم ode45 لحل المعادلة التالية للتابع y(t) ضمن المجال t = [0,10] مع شرط ابتدائي $y(0)=1-$

و مع العلم ان

$$dy/dt = -ty/10$$

ومن ثم ارسم الخرج.

الحل:

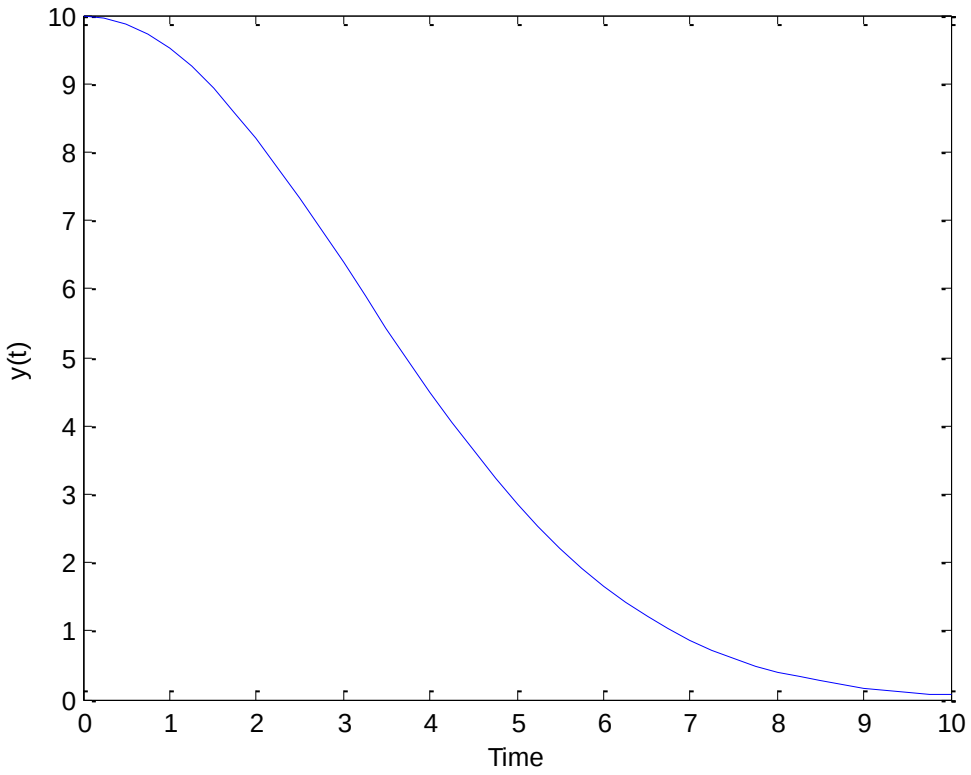
انشأ ملف تابع على الشكل:

```
function dydt=odefun(t,y)
dydt=-t*y/10;
```

ثم ضمن ملف script

```
[t,y]=ode45('odefun',[0 10],10);
 plot(t,y);xlabel('Time');ylabel('y(t)');
```

الخرج على الشكل التالي:



يمكن استخدام anonymous function على الشكل التالي:

```
[t,y]=ode45(@(t,y) -t*y/10,[0 10],10);
```

ملاحظة: يمكن ضمن MATLAB الحصول على حلول المعادلة التفاضلية بدلالة الرموز وذلك باستخدام التعليمة dsolve، الشكل القواعدي للتعليمة هو:

```
dsolve('eqn')
```

حيث 'eqn' هي عبارة عن معادلة مكتوبة كسلسلة من المحارف string.

تعيد التعليمة الحل بدلالة الرموز symbolic solution إضافةً إلى ثوابت اختيارية هي .... ,C1, C2. ومن الممكن أيضاً تحديد شروط ابتدائية أو حديّة للمسألة وفق الشكل التالي:

```
dsolve('eqn','cond1', 'cond2', …)
```

من أجل استخدام التعليمة dsolve بشكل صحيح، يتم التعبير عن المشتقات باستخدام المحرف D، مثلاً للتعبير عن المعادلة التالية:

$$f'(t) = -2 * f + \cos(t)$$

نكتب ما يلي:

```
'Df = -2*f + cos(t)'
```

أما من أجل تحديد مشتقات من مرتبة أعلى يجب أن يتبع رقم المرتبة حرف D أي للتعبير عن المعادلة التالية:

$$f''(t) + 2 * f'(t) = 5.sin(3t)$$

نكتب ما يلي:

'D2y+2Dy = 5*sin(3t)'

مثال: لنقم بحل المعادلة التفاضلية التالية $y' = 5y$ باستخدام MATLAB. نكتب الرماز التالي:

```
s = dsolve('Dy = 5*y')
```

الخرج هو:

```
C2*exp(5*t)
```

مثال: : لنقم بحل المعادلة التفاضلية التالية مع الشروط الابتدائية $y'' - y = 0, y(0) = -1, y'(0) =$ 2 باستخدام MATLAB. نكتب الرماز التالي:

```
dsolve('D2y - y = 0','y(0) = -1','Dy(0) = 2')
```

الخرج هو:

```
exp(t)/2 - (3*exp(-t))/2
```

التحويلات Transforms

يحتوي MATLAB على عدد من التحويلات المشهورة، والتي من الممكن تطبيقها بسرعة باستخدام تعليمة وحيدة مثل تحويل لابلاس Laplace وتحويل فوريه Fourier. سيتم التعرف على هذه التحويلات بشكل أكبر ضمن مقررات أخرى لذلك لن ندخل بالتفاصيل الرياضية ووجود التحويل وتقاربه، الهدف هو استخدام التعليمات للحصول على التحويلات.

1. تحويل لابلاس Laplcae Transform:

سنذكر فقط علاقة تحويل لابلاس:

$$LP\{f(t)\}(s) = \int_0^{+\infty} f(t).e^{-st}dt$$

التعليمة المستخدمة هي laplace.

مثال:

Editor - C:\Users\Simon\Documents\MATLAB\Laplace_tr.m

Laplace_tr.m

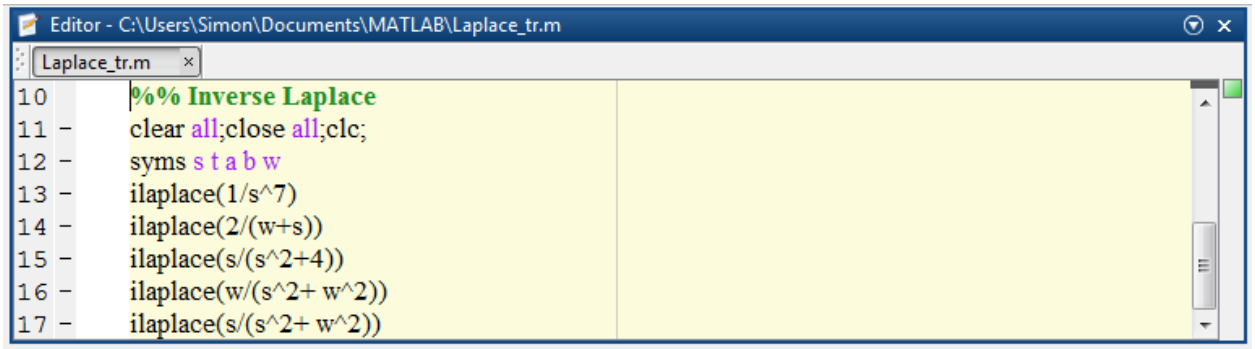
```
%% Laplace Transform
clear all;close all;clc;
syms s t a b w
laplace(a)
laplace(t^2)
laplace(t^9)
laplace(exp(-b*t))
laplace(sin(w*t))
laplace(cos(w*t))
```

الخرج هو:

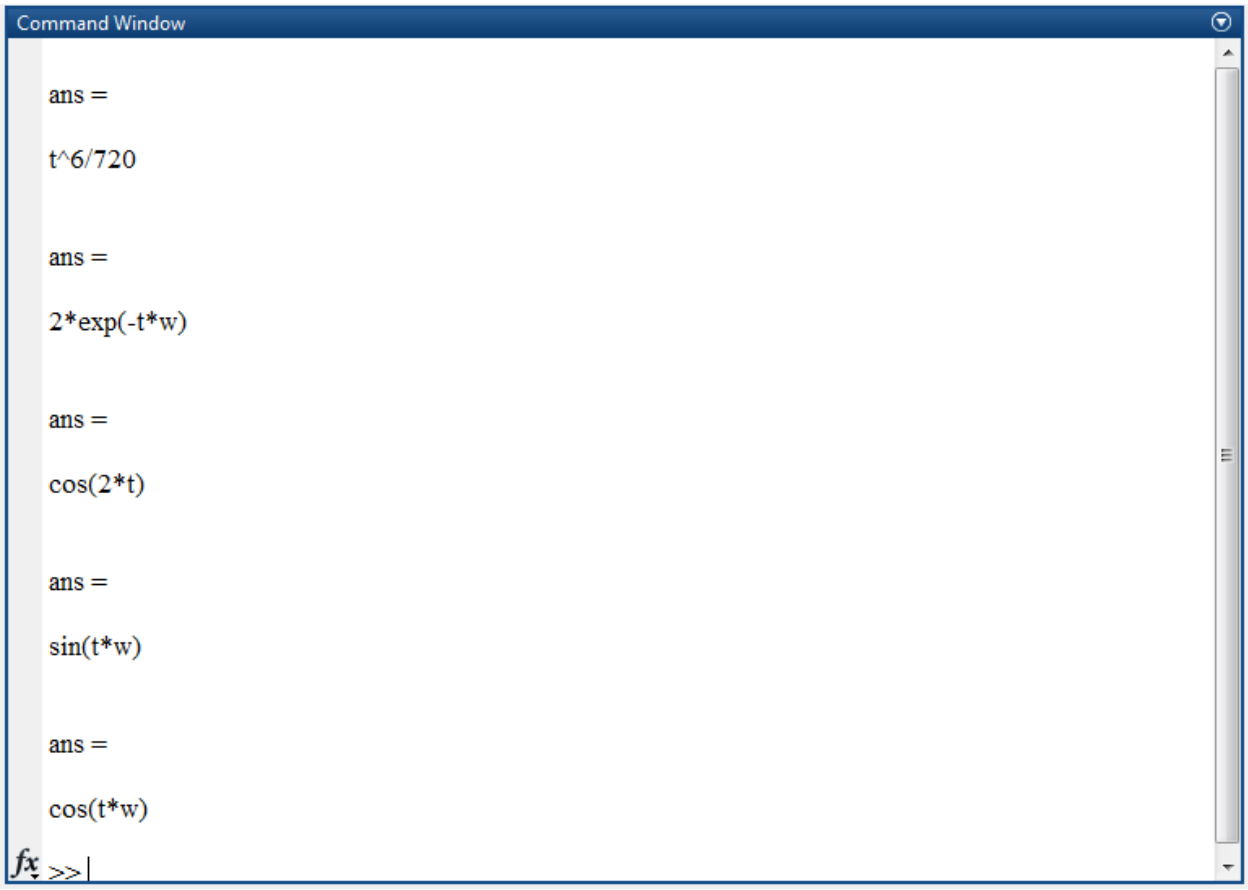
2. تحويل لابلاس العكسي Inverse Laplace Transform:

يتم عن طريق التعليمة ilaplace.

مثال:



الخرج هو:



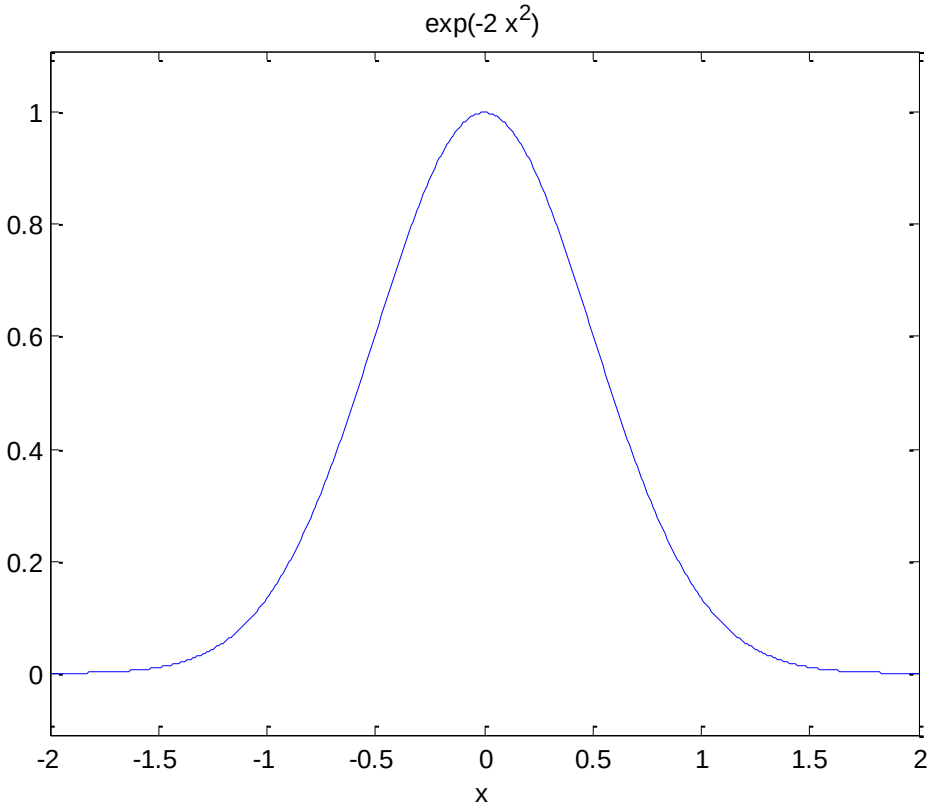
3. تحويل فوريه Fourier Transform:

يتم عبر استخدام تعليمة fourier.

مثال:

```
%% Fourier Transform
clear all;close all;clc;
syms x
f = exp(-2*x^2);%ourfunction
ezplot(f,[-2,2])% plot of ourfunction
FT = fourier(f)  %Fourier transform
```

الخرج هو:



إضافة للخرج التالي:

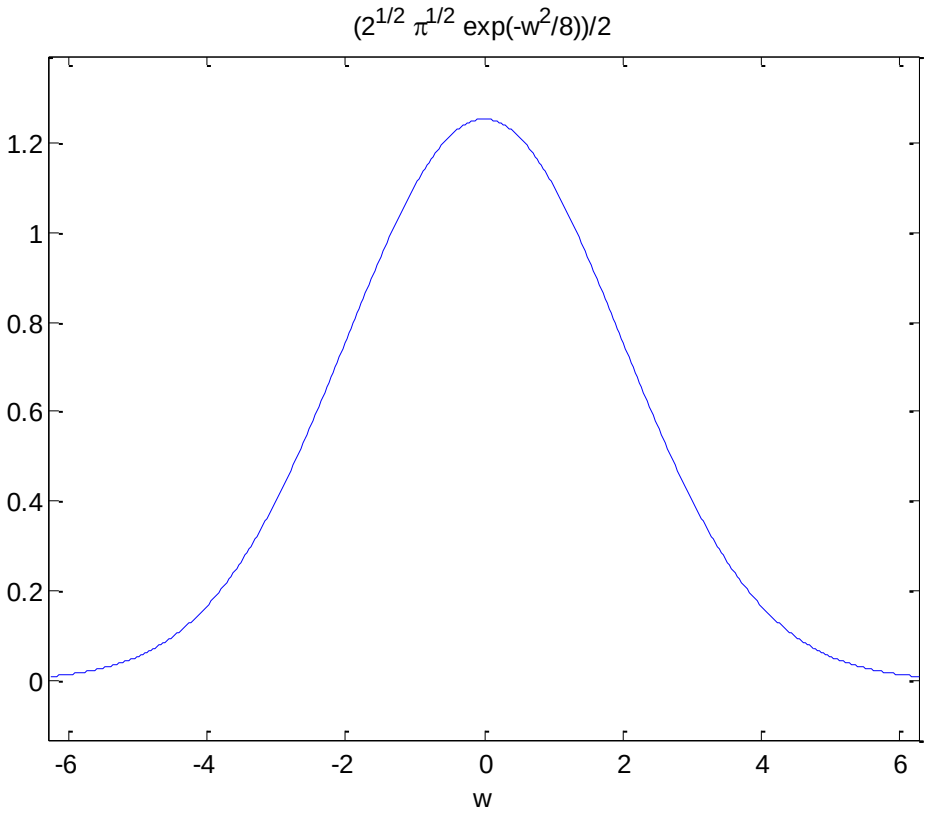
FT =

(2^(1/2)*pi^(1/2)*exp(-w^2/8))/2

ثم نقوم برسم تحويل فورييه باستخدام التعليمة التالية:

ezplot(FT)

فنحصل على الخرج التالي:



4. تحويل فوريه Fourier Transform:

يتم عبر استخدام تعليمة ifourier.

مثال:

```
%%
syms w
f = ifourier(-2*exp(-abs(w)))
```

الخرج هو:

```
f = -2/(pi*(x^2 + 1))
```

الاحتمالات والإحصاء Probability and Statistics

يعتبر علم الاحتمالات والإحصاء واحداً من أهم العلوم في دراسة الظواهر الطبيعية او دراسة العينات أو المعطيات وفهم سلوكها وتوزعها، نهدف في هذه الفقرة إلى إعطاء لمحة سريعة ومختصرة عن بعض التوابع المستخدمة في هذا المجال.

من أجل توليد معطيات عشوائية بتوزيع معين يمكن استخدام التعليمات التالية:

- rand, randn, randi

بعض التعليمات للحصول على المتوسط، الوسطي، ...:

- mean, median, mode
- min, max
- std, var
- pdf, cdf

الأسئلة

1. قم بكتابة رماز برمجي لحل جملة المعادلات التالية ومن ثم أظهر الحل ضمن Command Window.

$$6x + 10y = 15$$
$$8x - 6y = 5$$

مساعدة: قراءة فقرة حل المعادلات الجبرية

2. نرغب في حل المعادلة التالية بشكل يكون الخرج بدلالة الرمز، اكتب الرماز البرمجي اللازم المعادلة:

$$ax + b = 0$$

الخرج بدلالة المتحول x.

مساعدة: قراءة فقرة حل المعادلات الجبرية

3. قم بكتابة رماز برمجي لتعريف تابع لنحتولين بدلالة المتحولات كما في دروس النظري، التابع:

$$f(x, y) = x^2 + 12y - 5x + 1$$

مساعدة: قراءة فقرة حل المعادلات الجبرية

4. قم بكتابة رماز برمجي لتعريف كثير حدود على الشكل:

$$p(x) = x^9 - 17\,x^3 - 5x + 10$$

ومن ثم قم بإيجاد جذوره

مساعدة: قراءة فقرة كثيرات الحدود

5. قم بكتابة رماز برمجي لإيجاد نهاية التابع التالي عند القيمة 1 من اليمين واليسار ومن ثم رسمه على المجال [-5,5].

$$f(x) = \frac{x - 1}{|x - 1|}$$

مساعدة: قراءة فقرة النهايات

6. قم بكتابة رماز برمجي لإيجاد مشتق التابع التالي

$$f(x) = 5x^3 + 2x$$

ومن ثم إيجاد قيمة المشتق عند النقطة x=1.

مساعدة: قراءة فقرة الاشتقاق

7. قم بكتابة رماز برمجي لإيجاد تكامل التابع التالي:

$$5x^3$$

مساعدة: قراءة فقرة التكامل

8. قم بكتابة رماز برمجي لإيجاد قيمة تكامل التابع التالي على المجال [0,2*pi]

$$\sin(x) + 2\cos(x)$$

مساعدة: قراءة فقرة التكامل

الإجابات

1.
```
s = solve('6*x + 10*y = 15','8*x - 6*y = 5');
s.x
s.y
```

2.
```
syms a b x
solve(a*x+b == 0)
```

3.
```
syms x y real
syms f(x, y)
f(x, y) = x^2 + 12*y-5*x+1;
```

4.
```
%%
p = [1 0 0 0 0  0 -17  0 -5  10];
r = roots(p);
```

5.
```
syms x
f =(x -1)/abs(x-1);
ezplot(f,[-5,5])
L = limit(f,x,1,'left')
R = limit(f,x,1,'right')
```

6.
```
syms x
f =5*x^2+2*x;
diff(f)
diff(f,1)
```

.7

```
syms x
int(5*x^3)
```

.8

```
syms x
f = sin(x)+2*cos(x);
a =int(f,0,2*pi);
```

الفصل السابع: المحاكاة باستخدام Simulink®
Simulation using Simulink®

عنوان الموضوع:

المحاكاة باستخدام Simulink®

Simulation using Simulink®

الكلمات المفتاحية:

MATLAB، Simulink، معادلة تفاضلية Differential Equation، Solver، مكاتب Libraries، كتل Blocks، مولد إشارة Signal Generator، مكامل Integrator، مفاضل Derivative، منظار Scope، نواس بسيط Simple Pendulum، تعديل مطالي Amplitude Modulation، تابع تحويل Transfer Function.

ملخص:.

سنتعرف في هذا الفصل على جزء جديد ضمن برمجية MATLAB® يدعى Simulink®، وهو عبارة عن بيئة بيانية تساعد المستخدم في إجراء عملية محاكاة ونمذجة الأنظمة الديناميكية، سنتعرف أولاً على مبدأ المحاكاة ضمن هذه البيئة، وسنعرض ميزاتها. سيتم بعدها التعرف على مكاتب هذه البيئة والتي تتيح للمستخدم إنشاء الأنظمة انطلاقاً من الكتل الموجودة ضمنها، كما سنتطرق للتعرف على إعدادات المحاكاة لضمان الحصول على النتيجة الصحيحة ضمن حدود خطأ مقبولة يحدّدها المستخدم.

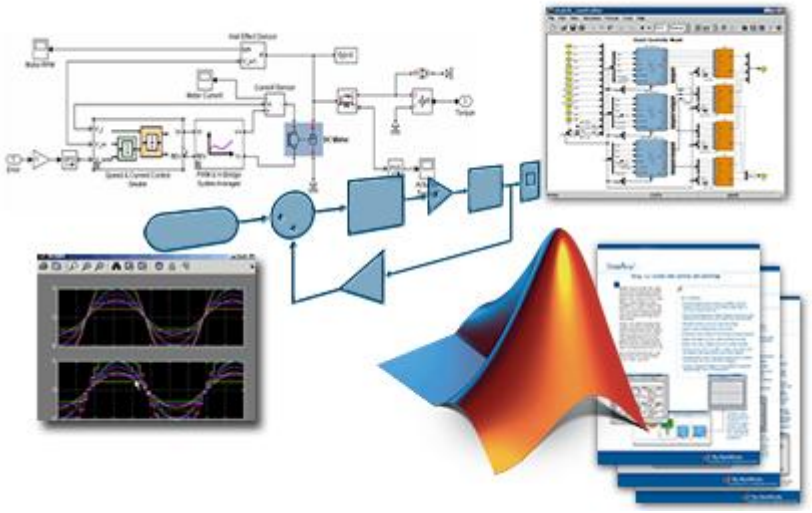
سنعرض بعد ذلك عدداً من الأمثلة التي تتناول مواضيع مختلفة بهدف توضيح آلية المحاكاة والتدرب على استخدام البيئة، ستشمل الأمثلة مواضيع الدارات الكهربائية، التعديل المطالي، الإشارات وبعض العمليات عليها، التعبير عن الأنظمة بدلالة المعادلات التفاضلية وتمثيل الأنظمة باستخدام توابع التحويل.

أهداف تعليمية:

يتعرف الطالب في هذا الفصل على:

- يتعرف على بيئة Simulink ضمن MATLAB.
- يستخدم المعادلات التفاضلية لتمثيل ونمذجة الأنظمة.
- ينجز محاكاة للعديد من الأنظمة وبطرق مختلفة باستخدام Simulink.
- يمتلك أداة قوية للمحاكاة تفيده في فهم الأنظمة المختلفة.
- يستوعب مواضيع مختلفة عن طريق نمذجة المشكلة بطريقة رياضية ومعاينة نتائج المحاكاة.

Introduction on Simulink® Environment — مقدمة عن بيئة Simulink®



®MATLAB، لغة الحوسبة التقنية، عبارة عن بيئة برمجية تهدف إلى تطوير الخوارزميات، تحليل المعطيات، التصوّر والحسابات العدّدية.

أما ®Simulink فهي عبارة عن بيئة بيانية تهدف إلى إجراء محاكاة وإنشاء تصاميم ونماذج Models لأنظمة ديناميكية و أنظمة مضمَّنة تغطي عدد كبير من المجالات.

يدعم جزء ®Simulink تصميم الأنظمة، المحاكاة، توليد الرمازات البرمجية بشكل أوتوماتيكي، وإجراء قياسات واختبارات للأنظمة المنفذة عملياً، لذلك فهو يستخدم في تصميم وتطوير مجال واسع من المنتجات المتقدمة عالية المستوى، مثلاً أنظمة ذاتية الحركة، أنظمة التحكم في الطيران وإلكترونيات الطيران، أنظمة الاتصالات، التجهيزات الالكترونية، الآلات الصناعية والأجهزة الطبية.

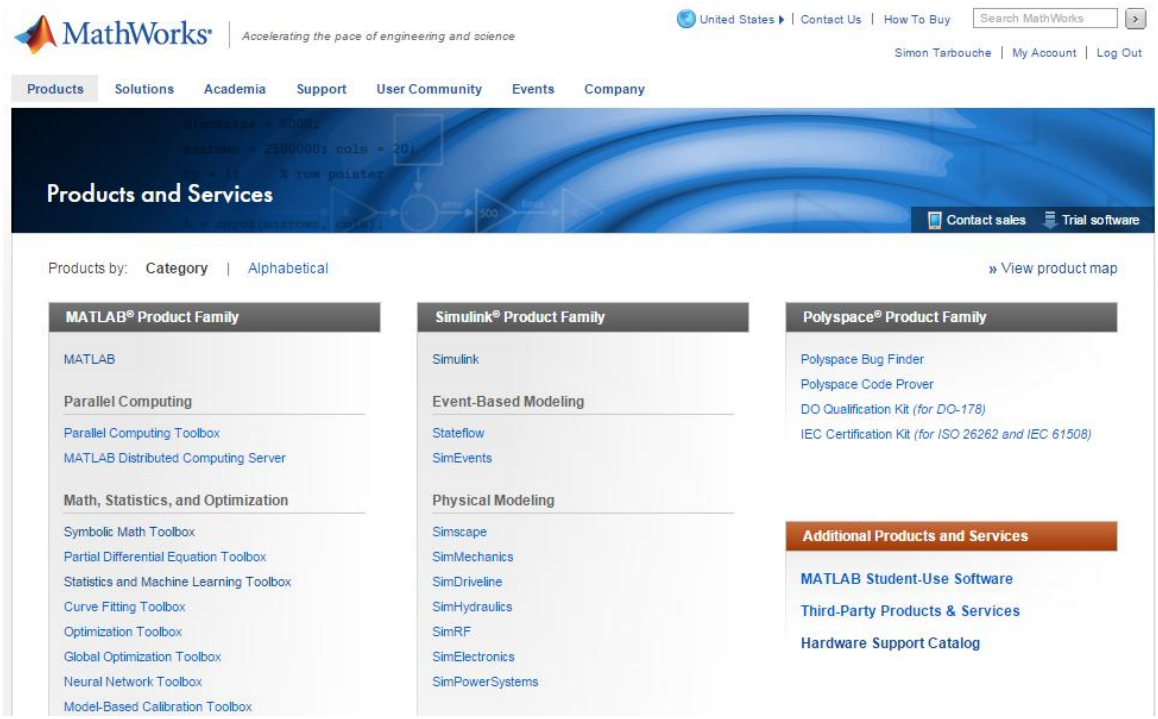
الميزات الرئيسية لـ Simulink® (Key Features):

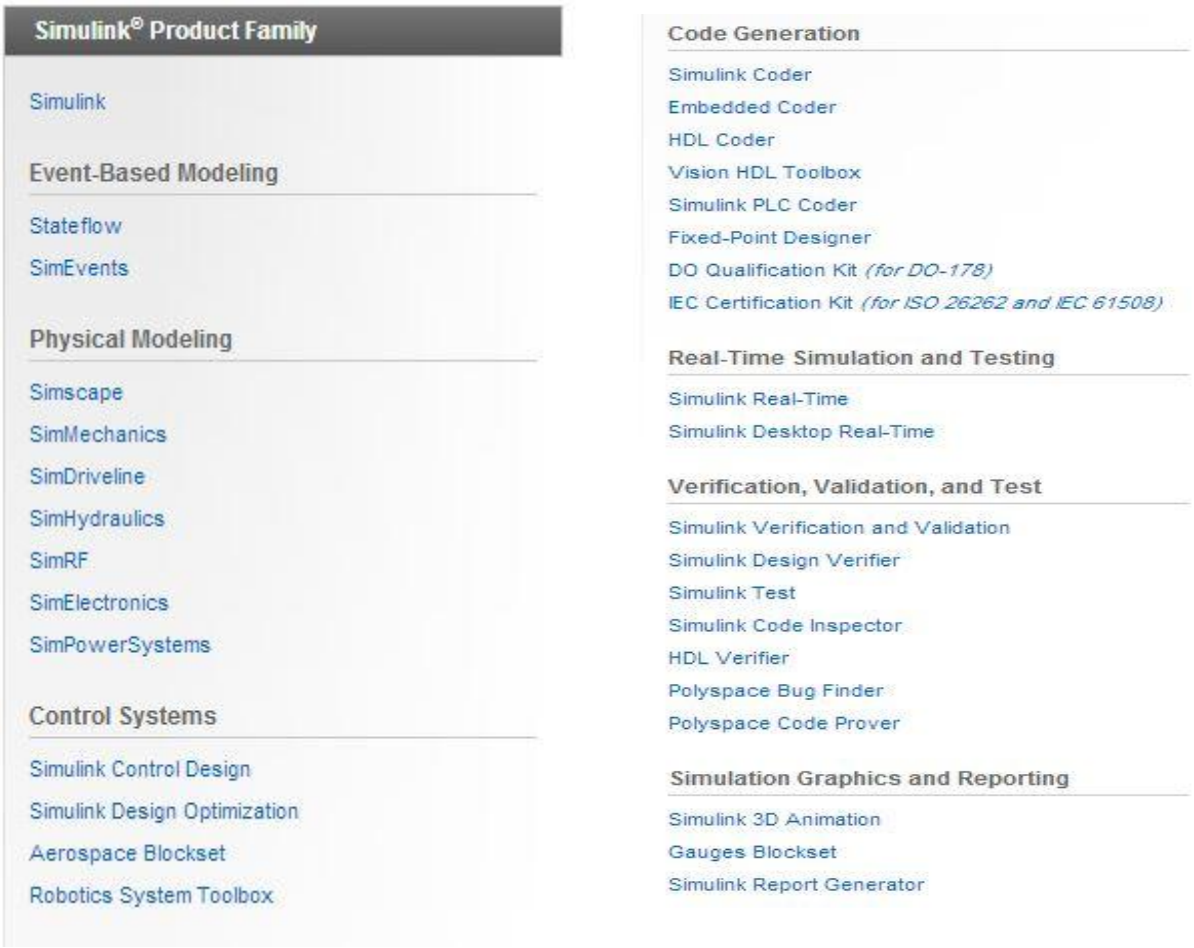
- محرّر بياني "Graphical editor" من أجل بناء وتنظيم المخططات الصندوقية block diagrams.
- مكاتب وكتل "Blocks" معرفة مسبقاً من أجل نمذجة الأنظمة المستمرة والمتقطعة والهجينة.
- محرك لإجراء المحاكاة باستخدام خطوات ثابتة أو متغيرة اعتماداً على المعادلات التفاضلية ODE Solvers.
- أدوات لعرض نتائج المحاكاة، مثل المنظار Scope.
- كتل وظيفية من أجل استيراد خوارزميات منجزة باستخدام لغة MATLAB إلى جزء Simulink®.

يمكن الاطلاع على جميع المنتجات المتعلقة بجزء Simulink® والمتوفرة ضمن برمجية MATLAB، عن طريق زيارة الموقع الالكتروني لشركة MathWorks® والدخول إلى Products.

http://www.mathworks.com/

الصور التالية تعرض بعض المنتجات والخدمات المتعلقة ب Simulink التي تؤمنها شركة MathWorks® وذلك ضمن مجالات مختلفة وتطبيقات متعدّدة.





البدء في استخدام Simulink® Start of using Simulink®

إن بيئة Simulink® تمت مكاملتها مع بيئة MATLAB® مما يتيح للمستخدم استخدام الخوارزميات المنجزة ضمن MATLAB وإضافتها إلى التصاميم أو النماذج Models المنجزة باستخدام Simulink®، إضافةً لإمكانية تصدير نتائج المحاكاة ضمن Simulink® إلى MATLAB وفضاء العمل Workspace والتعامل معها على أنها متحولات مما يتيح خيارات إضافية للمستخدم كإجراء تحليل للنتائج.

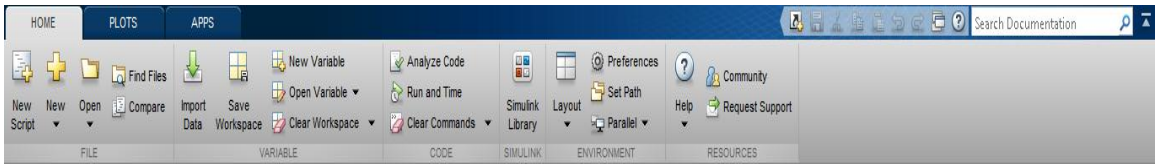
ذكرنا أن Simulink® يؤمن تخاطباً مع المستخدم عبر:

1. Graphical Editor محرر بياني من أجل بناء النماذج Model للأنظمة، يتميز بسهولته كما أنه تفاعلي مع المستخدم وهو ما يعرف ب Simulink Editor يتم بناء النماذج ضمنه عن طريق سحب الكتل المطلوبة من المكاتب الخاصة بها ووضعها ضمنه.
2. Block libraries مكاتب تحتوي على كتل كل منها يؤدي وظيفة معينة تشكل العناصر الأساسية في نمذجة أي نظام وهي موجودة ضمن ما يعرف ب Simulink Library Browser.

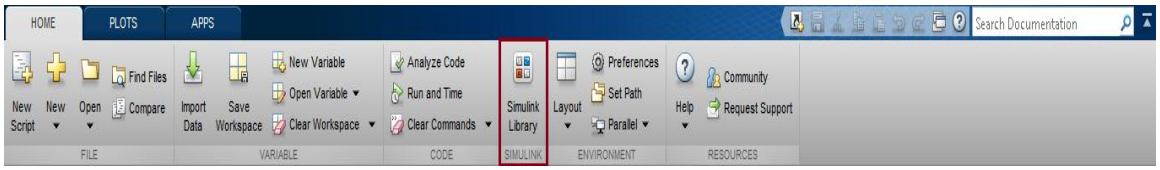
فتح متصفح المكاتب ضمن Simulink Open the Simulink Library Browser

من أجل فتح جزء Simulink® يمكن كتابة ضمن Command Window التعليمة simulink.

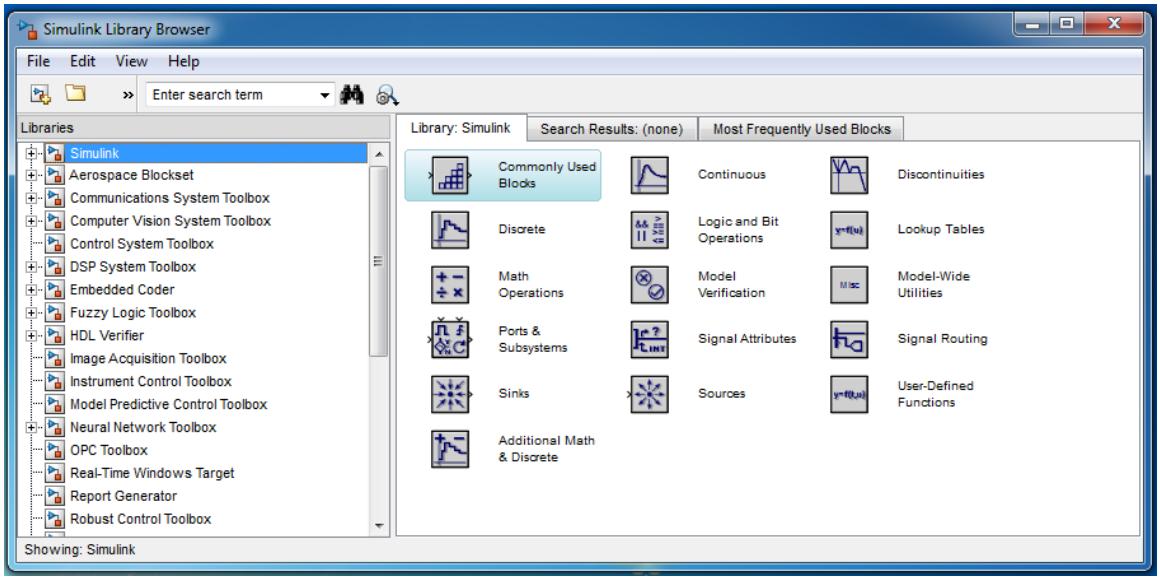
من أجل الوصول إلى Simulink Library Browser ضمن برمجية MATLAB، يمكن استخدام Toolstrip



ضمن Home الموجودة Toolstrip اضغط على Simulink Library



ستجد الواجهة التالية التي تحتوي على المكاتب الخاصة بـsimulink والتي جرى تعريفها مسبقاً من قبل مطوري البرمجية.



إنشاء نموذج Simulink جديد Create a New Simulink Model

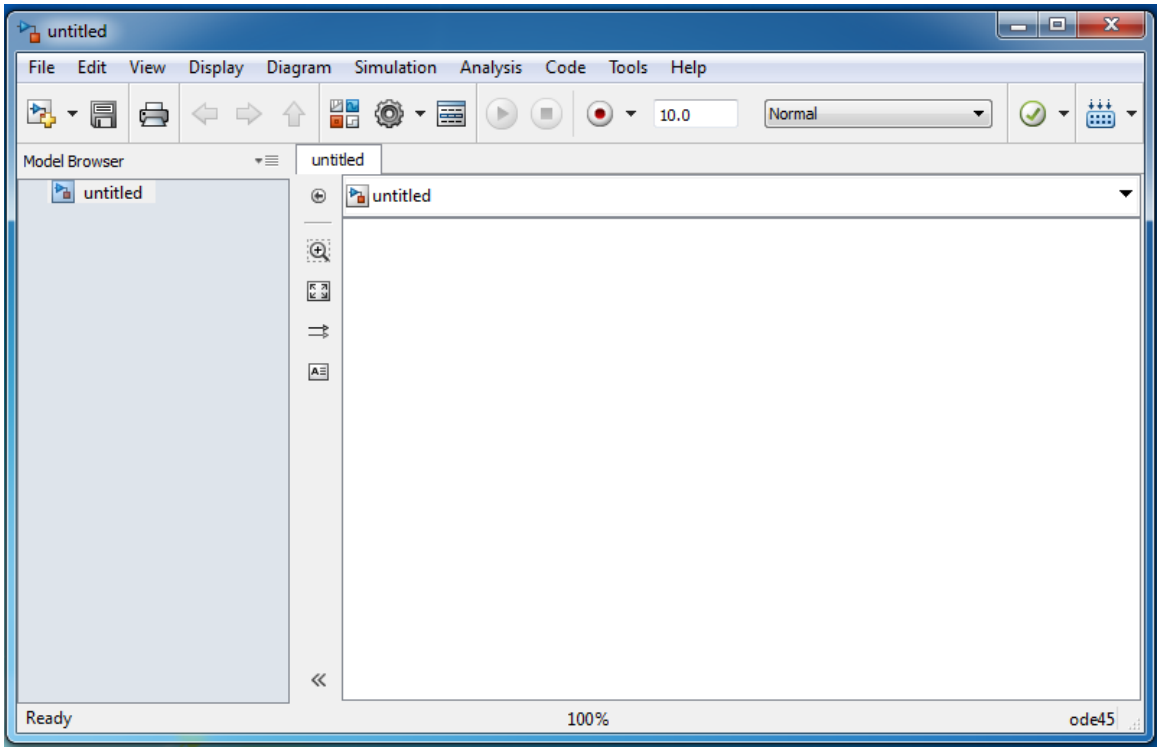
ويمكن أيضاً إنشاء Simulink Model من أجل تصميم النظام المراد إيجاد النموذج Model الموصّف له أو إجراء المحاكاة له، وذلك ضمن Home الموجودة Toolstrip اضغط على New

ستظهر أمامك لائحة، اختر منها Simulink Model، سيظهر أمامك الواجهة التالية.

أو بعد فتح Simulink Library Browser اضغط على file ومن ثم New ومن ثم Model، أو يمكن الضغط على الزر .

ضمن محرّر Simulink من اجل حفظ الملف اضغط على زر save أو من file ثم save، عندها لديك خيارين للحفظ كملف Simulink Models إما بلاحقة slx. أو mdl. يفضل الحفظ باستخدام اللاحقة mdl. حيث تظهر اللاحقة slx. بعض المشاكل نتيجةً للنسخة المتوفرة لدينا.

من أجل فتح ملف موجود مسبقاً، يمكن من file ثم open.



مبدأ عمل Simulink Principle of Simulink Work

ذكرنا أن Simulink يستخدم في نمذجة Modeling، محاكاة Simulation، وتحليل Analysis الأنظمة الديناميكة حيث يكون الخرج متغير مع الزمن وأنظمة التحكم الخطية واللاخطية منها إضافةً لأنظمة معالجة الإشارة،

تقسم عملية المحاكاة ضمن Simulink إلى جزئين رئيسيين ،لأول، يقوم المستخدم خلاله بإنشاء مخطط صندوقي للنظام المراد محاكاته وذلك باستخدام Simulink Editor وعن طريق كتل من المكاتب الموجودة ضمن Simulink Library Browser يتم عبر هذا الجزء توصيف العلاقات الرياضية المعتمدة على الزمن بين مداخل Inputs النظام، مخارجه outputs و حالاته states المختلفة. في المرحلة الثانية يعطي المستخدم الأوامر للبرمجية ببدء المحاكاة بعد تحديد زمن البداية والنهاية للمحاكاة إضافةً لاختيار Solver مناسب للحل.

ما هو Solver ؟؟؟

Solover هو عبارة عن عنصر ضمن بيئة Simulink، يقوم بتحديد الزمن للخطوة step اللاحقة ضمن المحاكاة وتطبيق طريقة عدّدية ما من أجل حل مجموعة من المعادلات التفاضلية الاعتيادية Ordinary Differential Equations ODE والتي تعبر عن نموذج Model النظام المراد محاكاته.

بعد ذلك يمكن المقايضة بين اختيار خطوة ثابتة Fixed-Step أو خطوة متغيرة Variable_Step وذلك من أجل القيم التي يجري محاكاة عندها، إضافةً لاختيار Solver مناسب حيث تختلف وفقاً لدرجة المعادلة التفاضلية أو المشاكل التي تسخدم لحلها.

يمكن للاطلاع أكثر على هذه المواضيع وفهم أكبر لمبدأ العمل الدخول إلى help وفق المسار التالي:

Simulink → Simulation → Configure Simulation → Choose a Solver

نجد مما سبق، أن الفكرة الرئيسية هو حل المعادلة التفاضلية، أو المعادلات، الموصفة للنظام. تكمن الحاجة للحل العدّدي من حقيقية عدم وجود حل تحليلي لجميع المعادلات التفاضلية وخاصةً اللاخطية منها.

الفكرة الأساسية في حل المعادلات التفاضلية هو تجزئة المعادلة إلى مجالات زمنية صغيرة ومن ثم حساب الحل العدّدي على المجالات السابقة، هذه المجالات هي حجم الخطوة step size التي يجري إجراء مقايضة trade off عند اختيارها بين fixed أو variable أي خطوة ثابتة أو متغيرة، يحتوي Simulink على عدد كبير من solvers، وننوه دوماً ان الحسابات العدّدية تقترن بوجود أخطاء لذلك فزيادة مرتبة المعادلة التفاضلية يساعد في التقليل من الأخطاء إلا أنه يستهلك الكثير من الوقت لإنهاء المحاكاة بسبب زيادة تعقيد خوارزمية الحل العدّدي، على المستخدم إجراء المقايضة بين الزمن الكبير للمحاكاة والدقة عند اختياره solver.

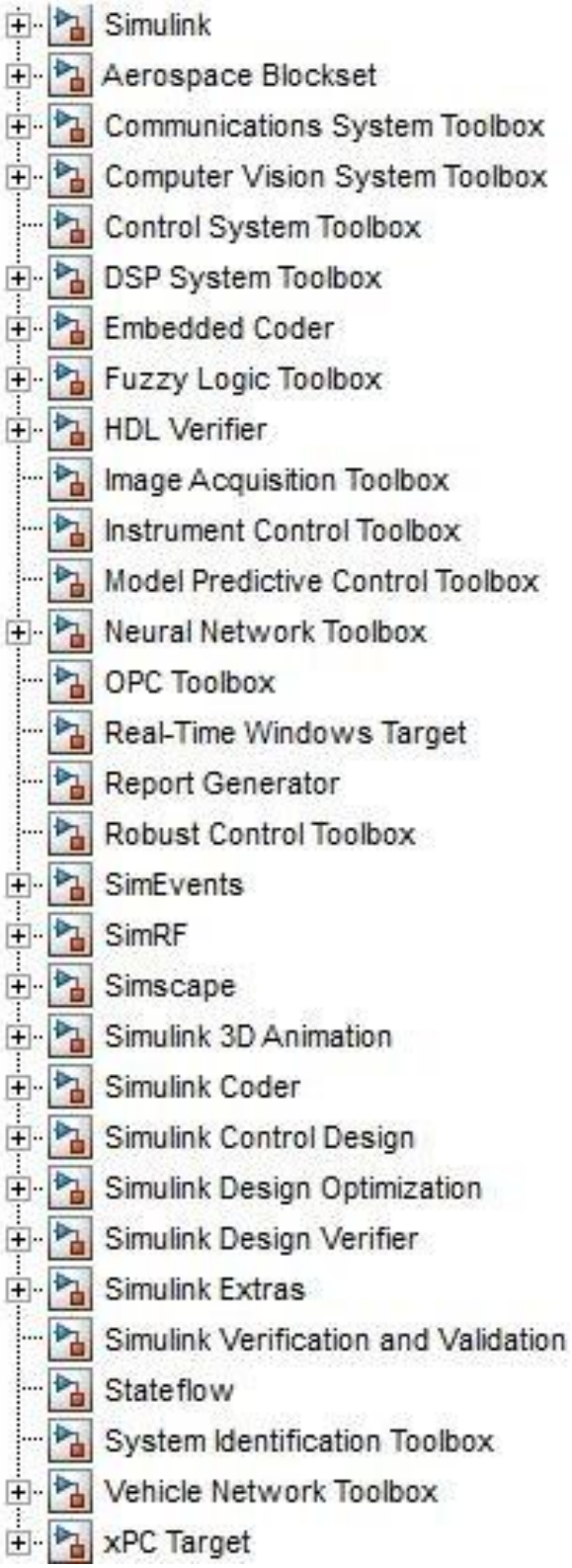
التعرف أكثر على مكاتب Simulin getting to know more about simulink libraries:

ما هي الكتل what are blocks؟؟

الكتل هي العناصر التي نستطيع باستخدامها بناء النماذج والتصاميم ضمن بيئة Simulink، حيث أن تمثيل النظام يكون عبارة عن إيجاد وصل بين كل كتلة بحيث تعبر كل كتلة عن وظيفة معينة أو مكون ما ضمن النظام. تحتوي غالبية الكتل على متحولات يجري التحكم بها من أجل تخصيصي سلوك الكتلة، مثلاً ضمن كتلة مولد الإشارة Signal Generator يمكن تحديد نمط الإشارة كإشارة جيبية أو سن منشار أو مربعة، وتحديد مطال وتردّد الغشارة المولدّة وبهذا نحصل على على العنصر المناسب لإجراء اختبارنا أو معرفة سلوك النظام وفق فرضية ما أو من أجل دخل معين.

يوجد ضمن Simulink Library Browser عدد كبير من المكاتب التي تقسم ضمنها إلى مكاتب فرعية وكل مكتبة فرعية تحتوي على عدد كبير من الكتل، سنعرض بعض من الكتل المفيدة والمستخدمة في كثير من الأحيان.

توضح الصورة التالية المكاتب:



من اهم المكاتب:

Simulink, Communication System Toolbox, SimRF, Control System Tollbox, DSP System Toolbox, Simscape.

تختص Communication System Toolbox بجميع الكتل المفيدة في مجال هندسة الاتصالات وتشمل على العديد من المكاتب الفرعية هي:

- القنوات Channels.
- المرشحات Comm Filters.
- منابع ورواسم خاصة مثل مخطط العين Comm Sources and Sinks.
- مسويّات Equalizer.
- التعديلات Modulations حيث تحتوي المكاتب على التعديلات الرقمية digital والتمثيلية Analog.
- ترميز المنبع Source Coding.
- التزامن Synchronization.

إضافةً إلى عدد من المكاتب الفرعية الأخرى، يرجى الأخذ بعين الاعتبار أن هذه الكتل ستساعدك كثيراً في إجراء محاكاة لأنظمة اتصالات بطريقة سهلة وفعالة يرجى الاستفادة منها في المقررات والسنوات اللاحقة.

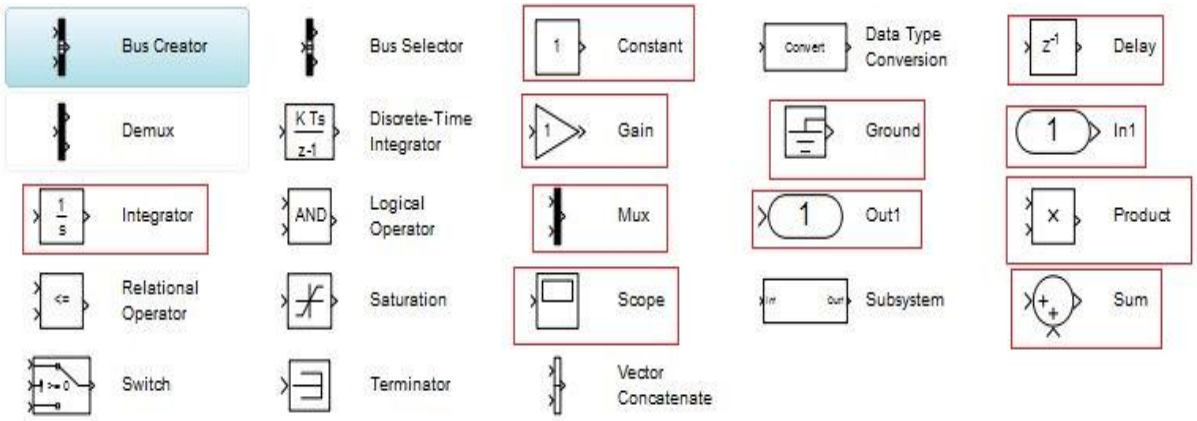
أما من أجل الكتل الخاصة بالإشارات الراديوية Radio Frequency وهندسة الأمواج المكروية Microwace Engineering فهذه الماكتب توجد ضمن SimRF.

تفيد Simscape في محاكاة عدد كبير من المواضيع في مجالات هندسة النظم الالكترونية، النظم الهيدروليكية، النظم الميكانيكية وأنظمة القدرة Power Systems.

سنتعامل كثيراً مع Simulink، تحتوي على عدد كبير من المكاتب الفرعية أيضاً والتي سنستخدمها سنعرض اهم المكاتب الفرعية المستخدمة في محاكاة الأنظمة المستمرة، والمتقطعة والهجينة.

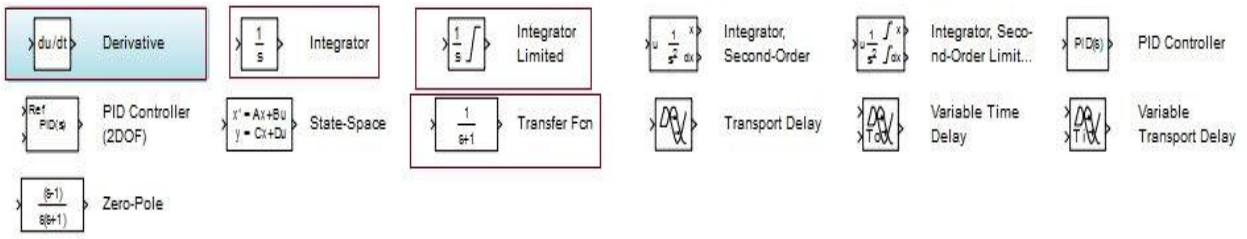
Commonly Used Block:

تحتوي على عناصر تستخدم كثيراً مثل المنظار scope، قيمة ثابتة Constant، مكامل Integrator، وحدة تأخير Delay، MUX والتي تفيد في تحديد مجرى للإشارات Signal Routing، ربح Gain، أرضي Ground، جداء Product، جمع Sum، دخل In، خرج Out.



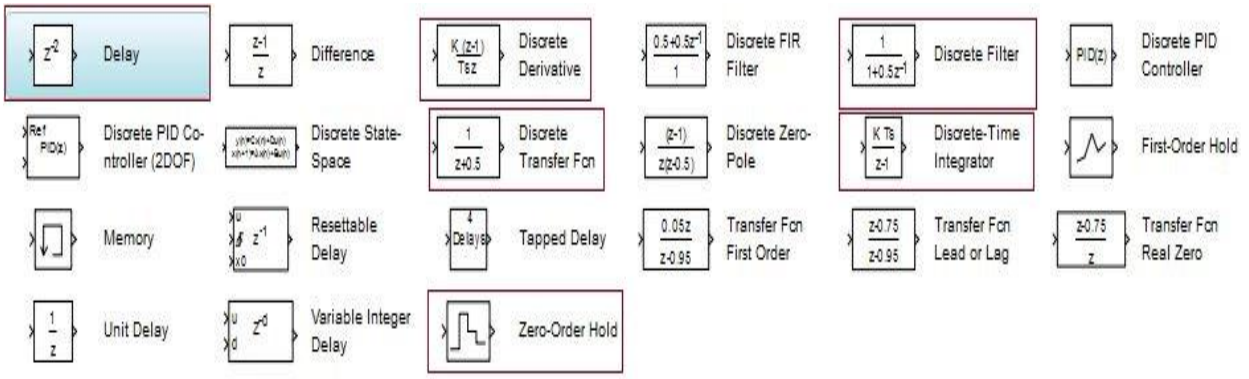
Continuos

مفاضل Derivative، مكامل Integrator، تابع تحويل Transfer Fcn، مكامل محدود Integrator Limited



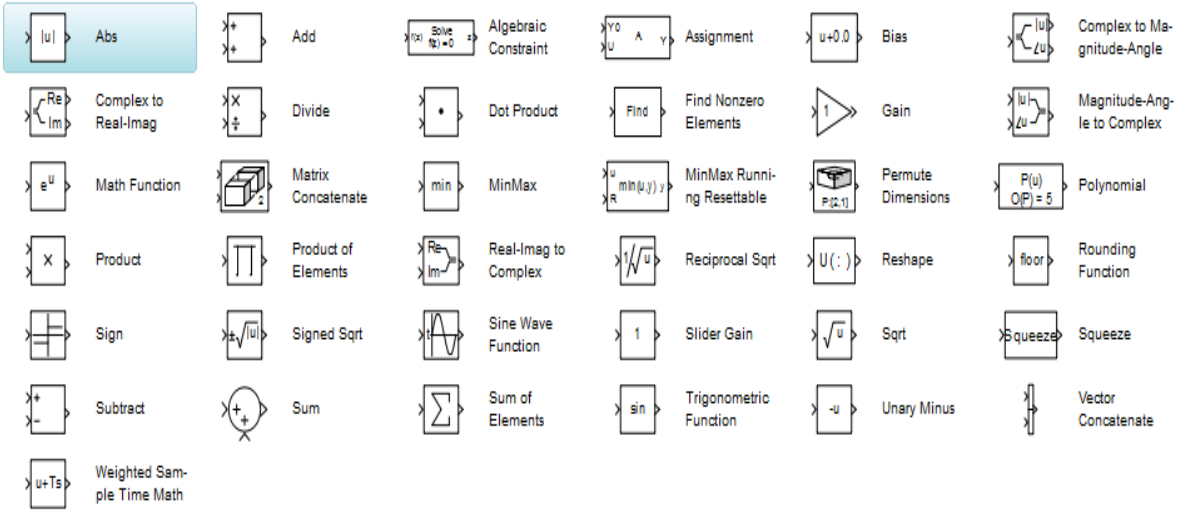
Discrete

تأخير Delay، مفاضل متقطع Discrete Derivative ، مرشح متقطع Discrete Filter ، تابع تحويل متقطع Discrete Transfer Fcn ، مكامل متقطع بالزمن Discrete-Time Integrator، ماسك Zero-Order



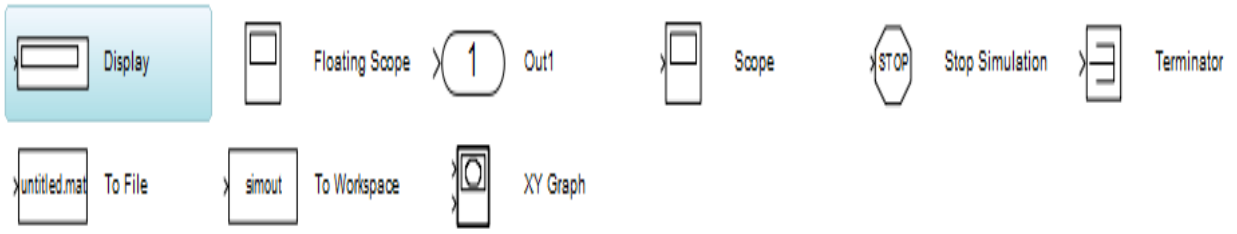
Math Operation

جميع العمليات مهمة ومفيدة نذكر منها: القيمة المطلقة، الجمع، الطرح، الجداء، القسمة، الربح، إشارة تابع، توابع جيبية، توابع رياضية، جداء سلمي، جذر، ....



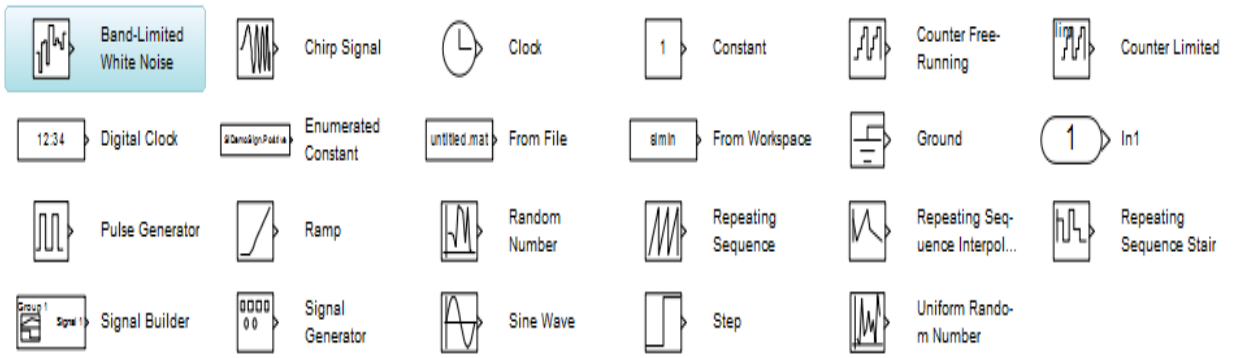
Sinks

تحتوي على كتل تفيد في الإظهار النتائج، إضافةً لذلك تحتوي على كتلتين مهمتين هما: To File لتصدير نتائج المحاكاة إلى ملف، To Workspace من اجل تصدير نتائج المحاكاة إلى فضاء العمل.

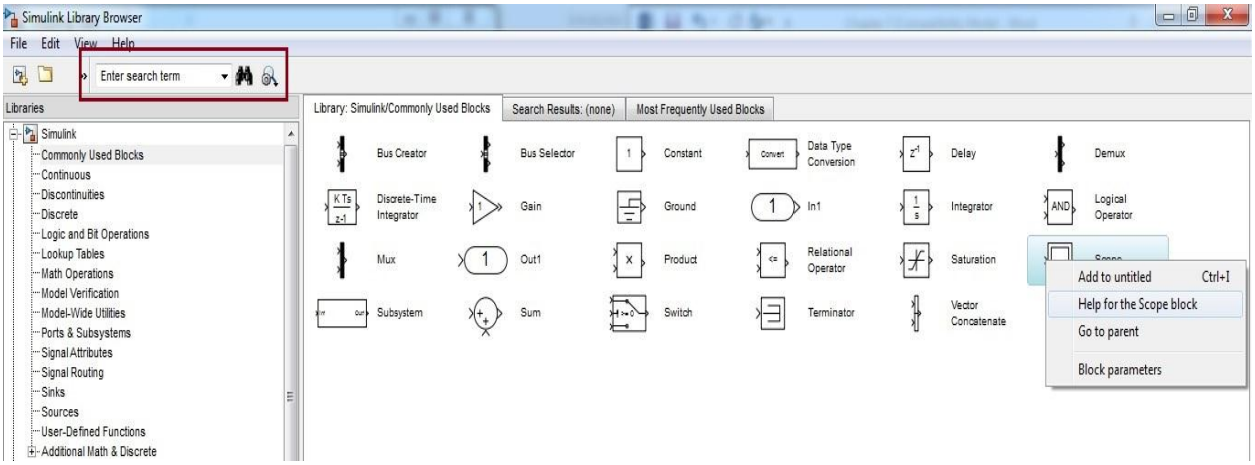


Sources

تحتوي على المنابع من اهمها Sine Wave، Step، Signal Generator.



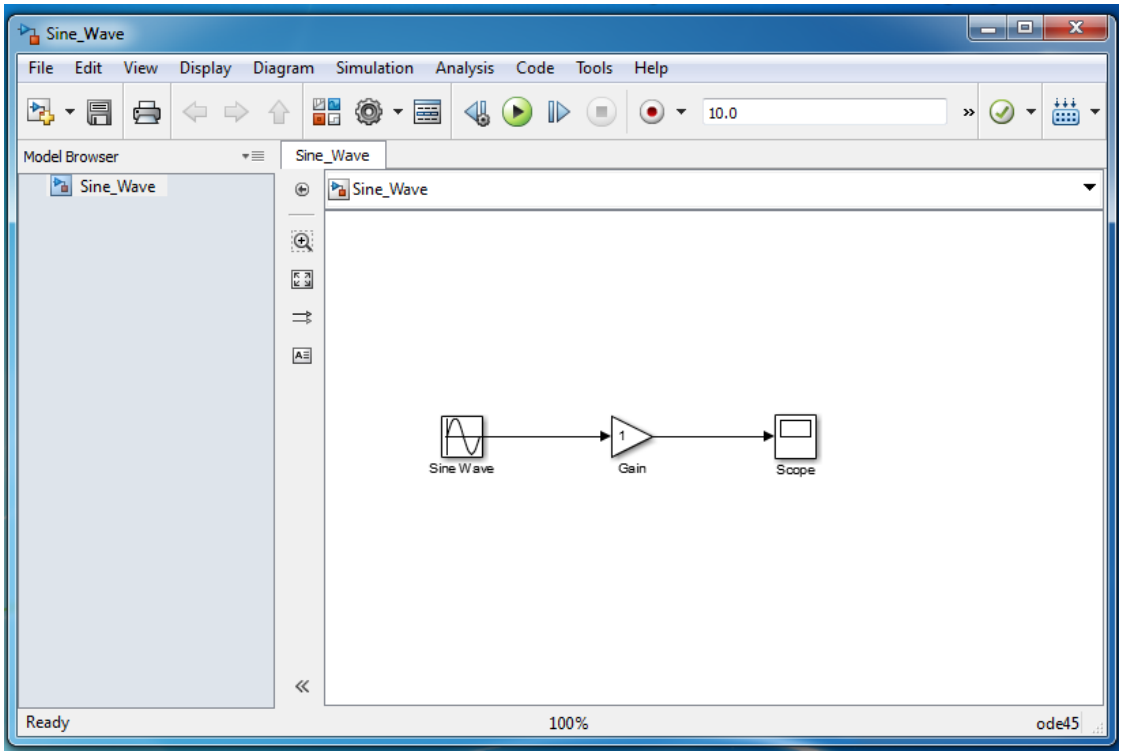
تتيح لنا Simulink Library Browser خيار البحث عن العنصر، يكفي فقط معرفة جزء من الاسم للكتلة، كما انه من المفيد كثيراً استخدام help المرافق لكل عنصر لمعرفة توصيفه بشكل دقيق. كما هو موضح في الشكل التالي، حدد الكتلة ومن ثم اضغط على الزر اليميني واختر help.



أمثلة Examples

مثال: سنقوم ضمن هذا المثال بإنشاء نموذج بسيط حيث سيتم توليد إشارة جيبية ومن ثم يتم تضخيمها بمقدار ربح معين يمكن التحكم به ومن ثم عرض الخرج. قم بإنشاء ملف جديد New Model وسمّه Sine_Wave.

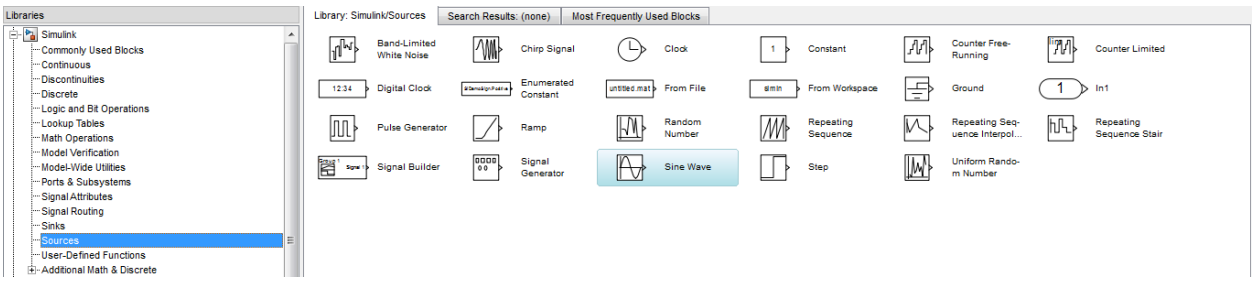
الشكل النهائي للدارة هو:



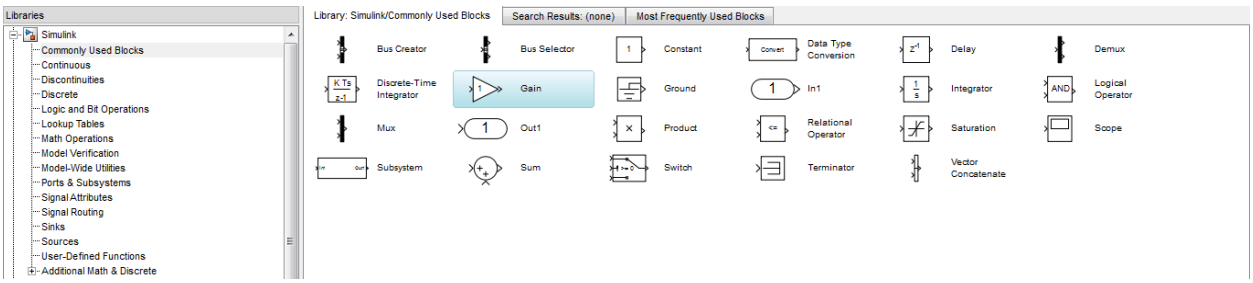
نحتاج إلى العناصر التالية:

- Sine Wave يستخدم من أجل توليد إشارة الدخل ضمن النموذج.
- Gain يستخدم من أجل تضخيم إشارة الدخل بمعامل ربح ما.
- Scope يستخدم من أجل معاينة الخرج.

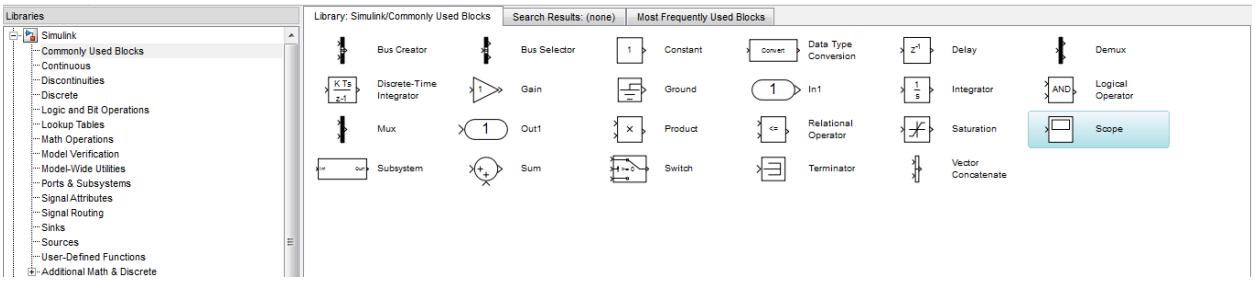
لإحضار كتلة Sine Wave، من قائمة المكاتب اذهب إلى Simulink ثم Sources ومن ثم اعثر على Sine Wave.



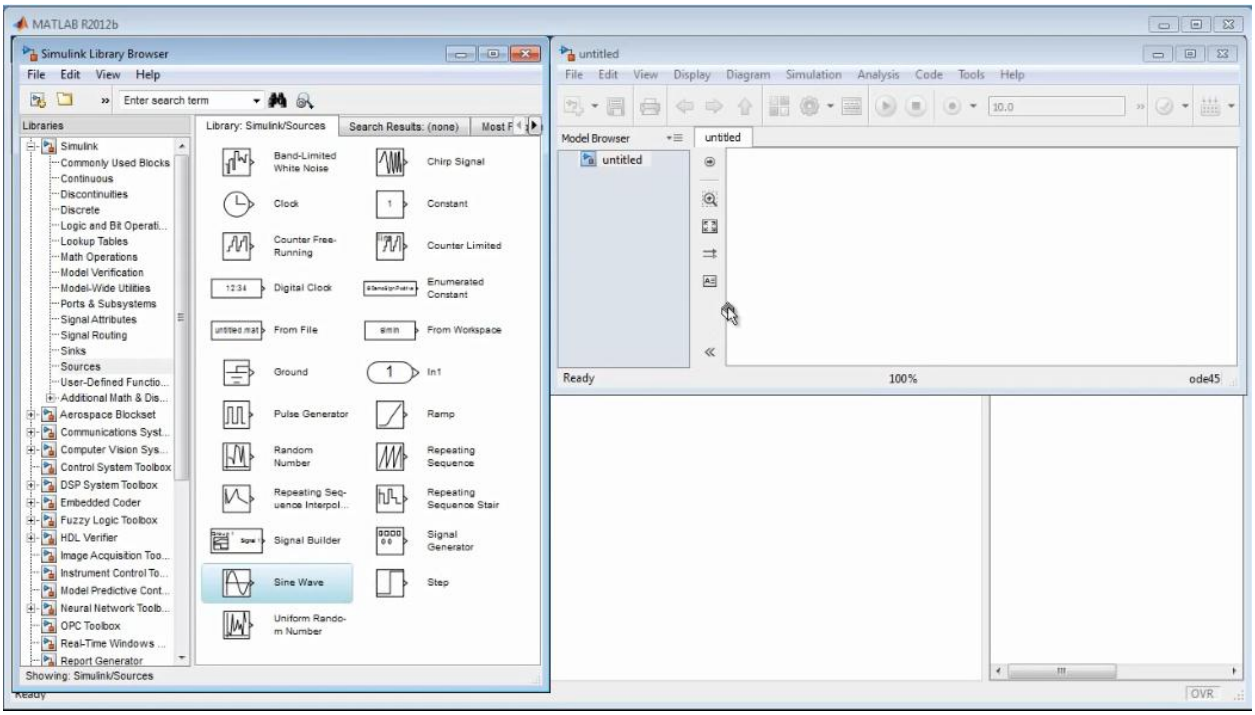
لإحضار كتلة Gain، من قائمة المكاتب اذهب إلى Simulink ثم Commonly Used Blocks ومن ثم اعثر على Gain.



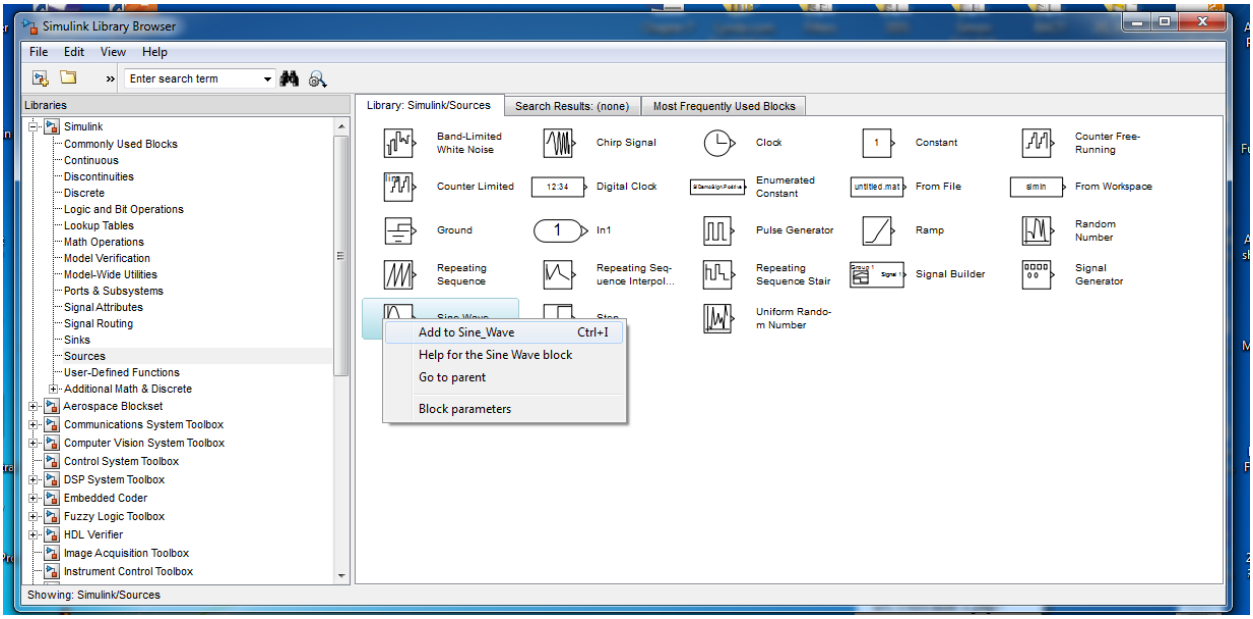
لإحضار كتلة Scope، من قائمة المكاتب اذهب إلى Simulink ثم Commonly Used Blocks ومن ثم اعثر على Scope.



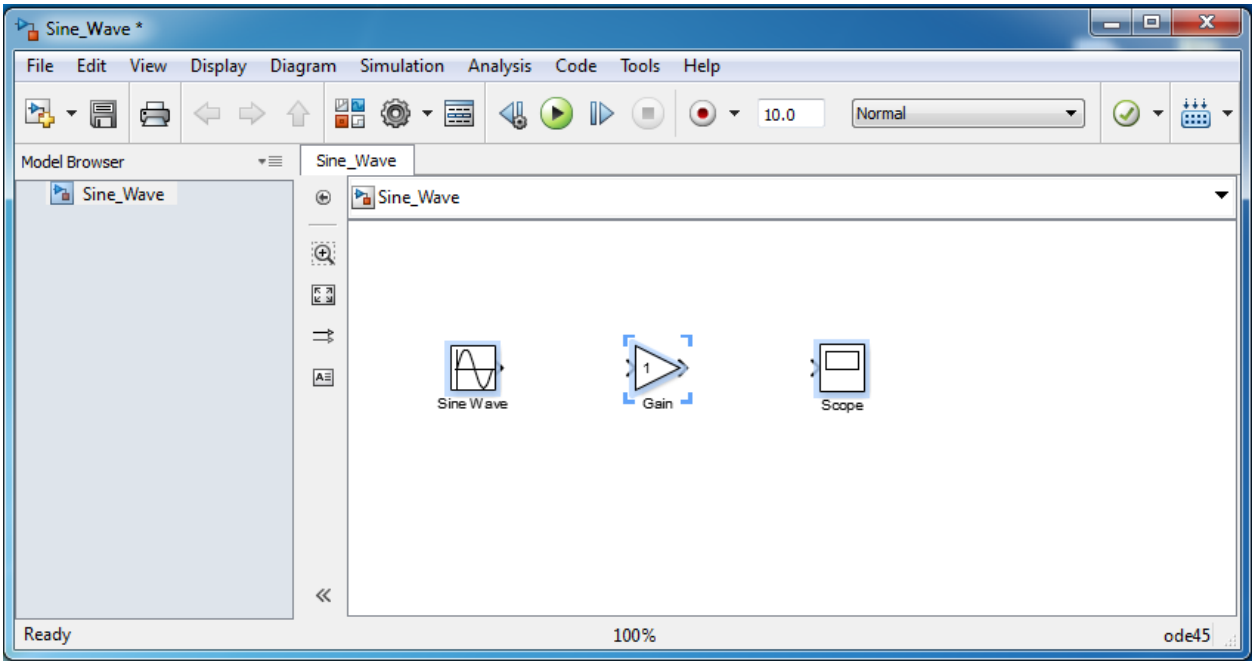
عند الضغط على الكتلة يمكن سحبها إلى الملف المنشأ من أجل المحاكاة ووضع الكتلة ضمنه كما في الشكل



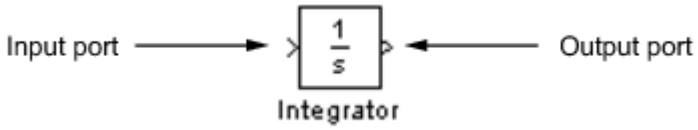
او اضغط على الكتلة بزر الفأرة اليميني واختر Add to Sine_Wave، طبعاً Sine_Wave حسب اسم الملف الذي قمت بتسميته عند الإنشاء، يوضح الشكل التالي ذلك.



عند وضع الكتل بشكل كامل تحصل على الشكل:



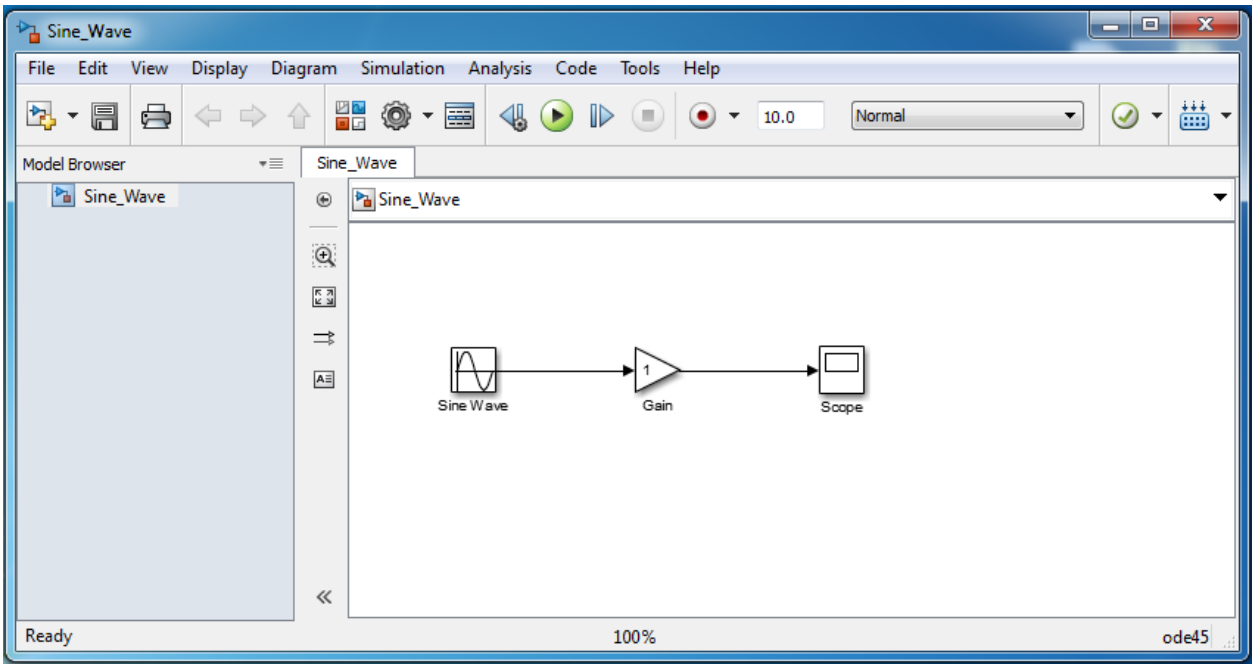
ملاحظة: عند جلب كتلة من المكتبة نلاحظ وجود سهمين على دخل وخرج (أحياناً سهم واحد فقط) تدل هذه الأسهم على دخل Input الكتلة و خرجها Output.



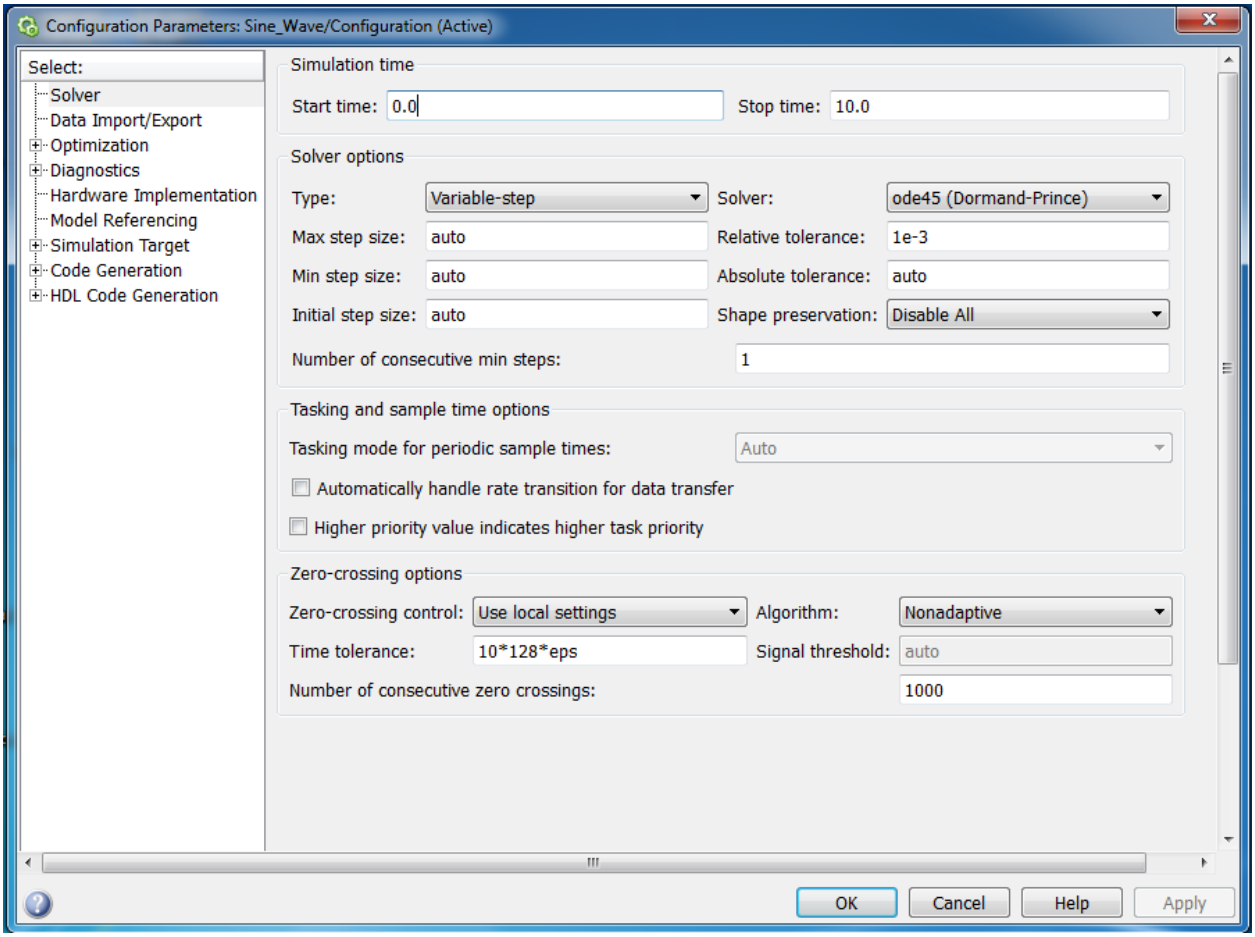
من أجل الحصول على الخرج ومشاهدته لابد لنا من الوصل بين الكتل باستخدام خطوط الإشارة Signal Lines، ببساطة يمكن رسم الخطوط عن طريق تحديد الكتلة ورسم الخط بين مخرج output الكتلة ومدخل

input الكتلة الأخرى، أو عن طريق الضغط على الكتلة الأولى ومن ثم الضغط على زر Ctrl باستمرار والضغط على الكتلة الثانية لتحصل على خط إشارة يصل بين العنصرين.

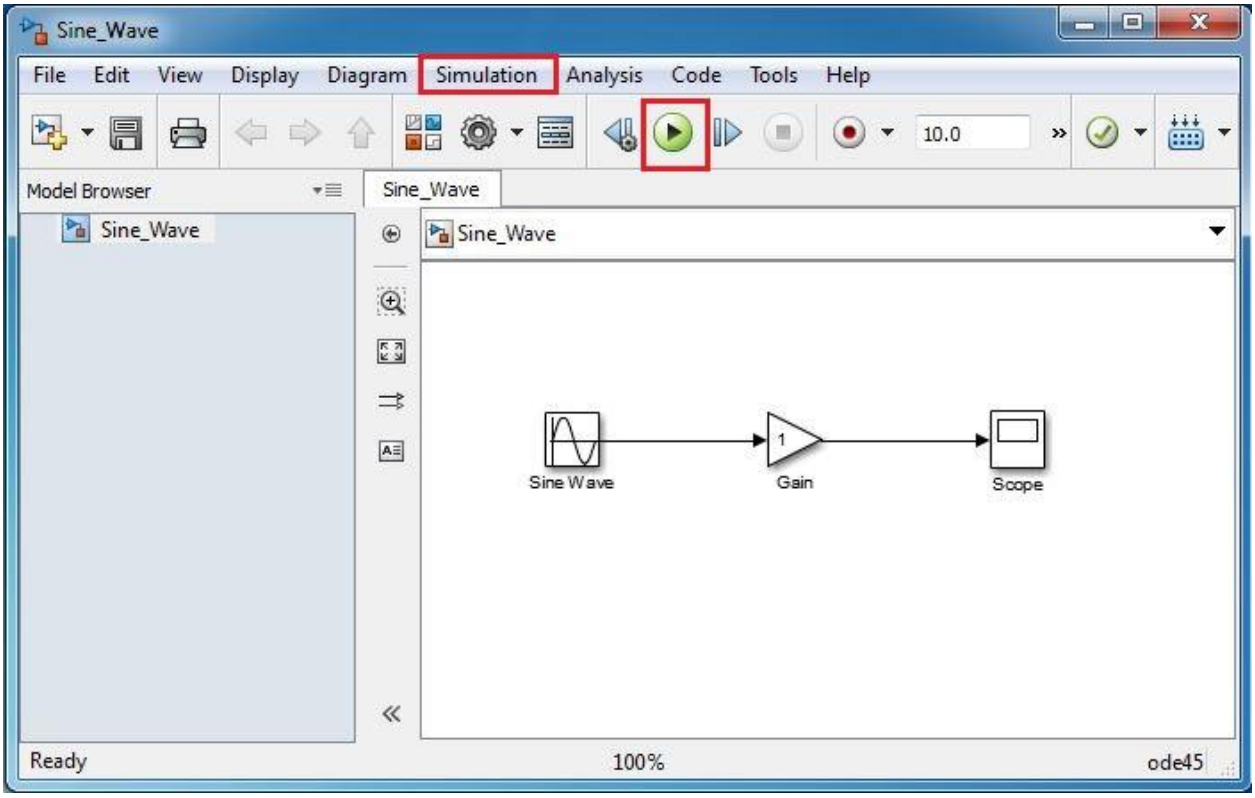
قم ببناء النموذج للحصول على الشكل النهائي التالي:



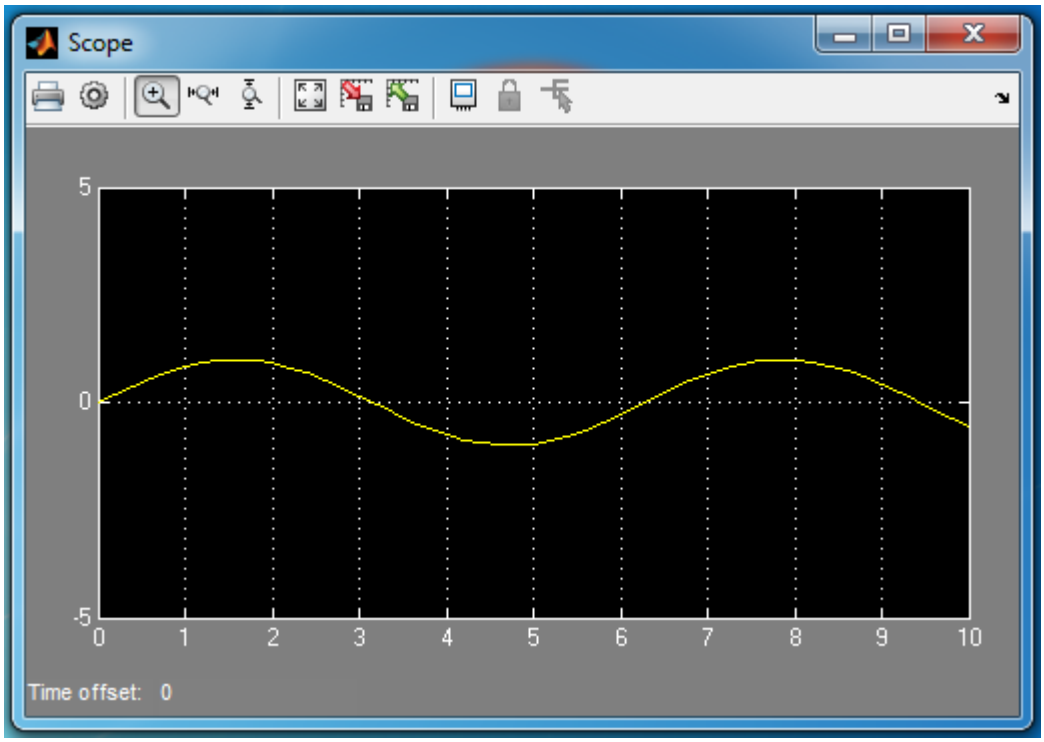
لنقم الآن بمحاكاة النموذج التالي وعرض النتائج وذلك دون التغيير بالإعدادات حيث أن الإعدادات الافتراضية هي:


Configuration Parameters: Sine_Wave/Configuration (Active)
Select:
Solver
Data Import/Export
Optimization
Diagnostics
Hardware Implementation
Model Referencing
Simulation Target
Code Generation
HDL Code Generation
Simulation time
Start time: 0.0
Stop time: 10.0
Solver options
Type: Variable-step
Solver: ode45 (Dormand-Prince)
Max step size: auto
Relative tolerance: 1e-3
Min step size: auto
Absolute tolerance: auto
Initial step size: auto
Shape preservation: Disable All
Number of consecutive min steps: 1
Tasking and sample time options
Tasking mode for periodic sample times: Auto
Automatically handle rate transition for data transfer
Higher priority value indicates higher task priority
Zero-crossing options
Zero-crossing control: Use local settings
Algorithm: Nonadaptive
Time tolerance: 10*128*eps
Signal threshold: auto
Number of consecutive zero crossings: 1000
OK
Cancel
Help
Apply

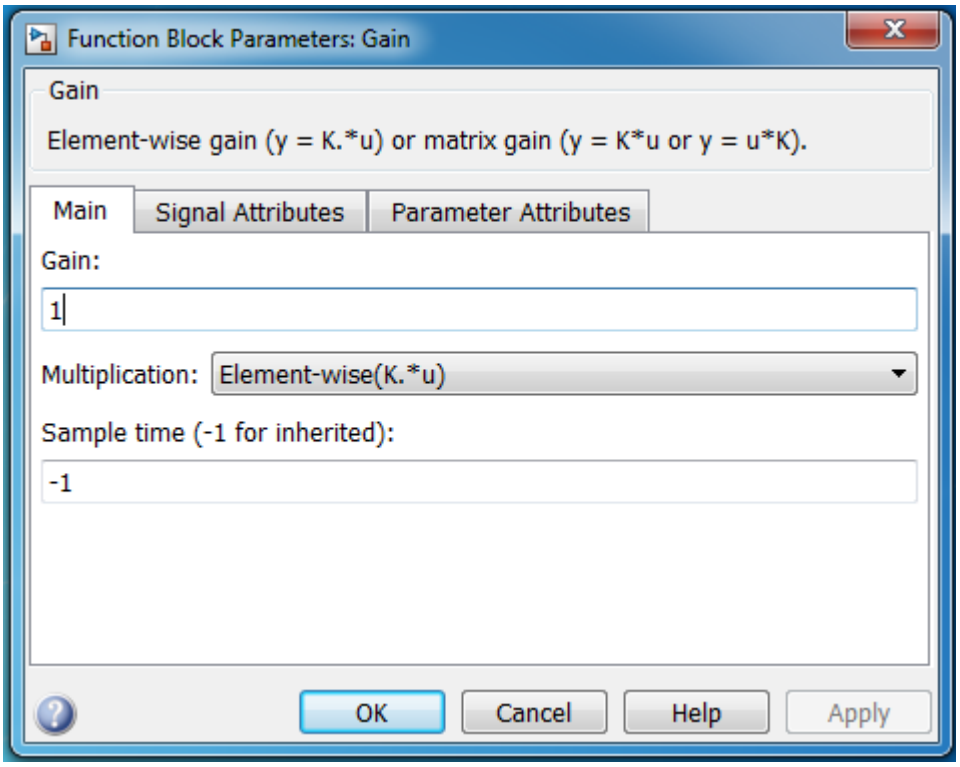
من أجل تشغيل المحاكاة اضغط على زر Run كما في الشكل أو من الشريط اختر Simulation ثم Run.



ثم اضغط على Scope مرتين لتفتح واجهة جديدة وتحصل على النتيجة التالية:

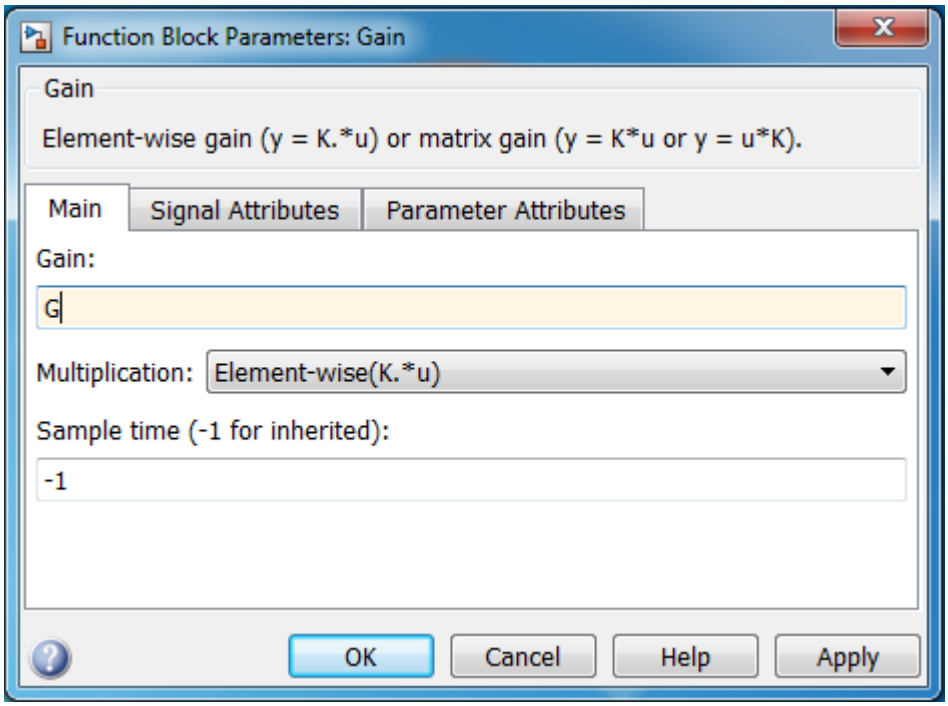


استخدم aotuscale  لمشاهدة الخرج بشكل أوضح. سنقوم الآن بتغيير قيمة التضخيم للإشارة، اضغط على كتلة Gain ستحصل على النافذة التالية:



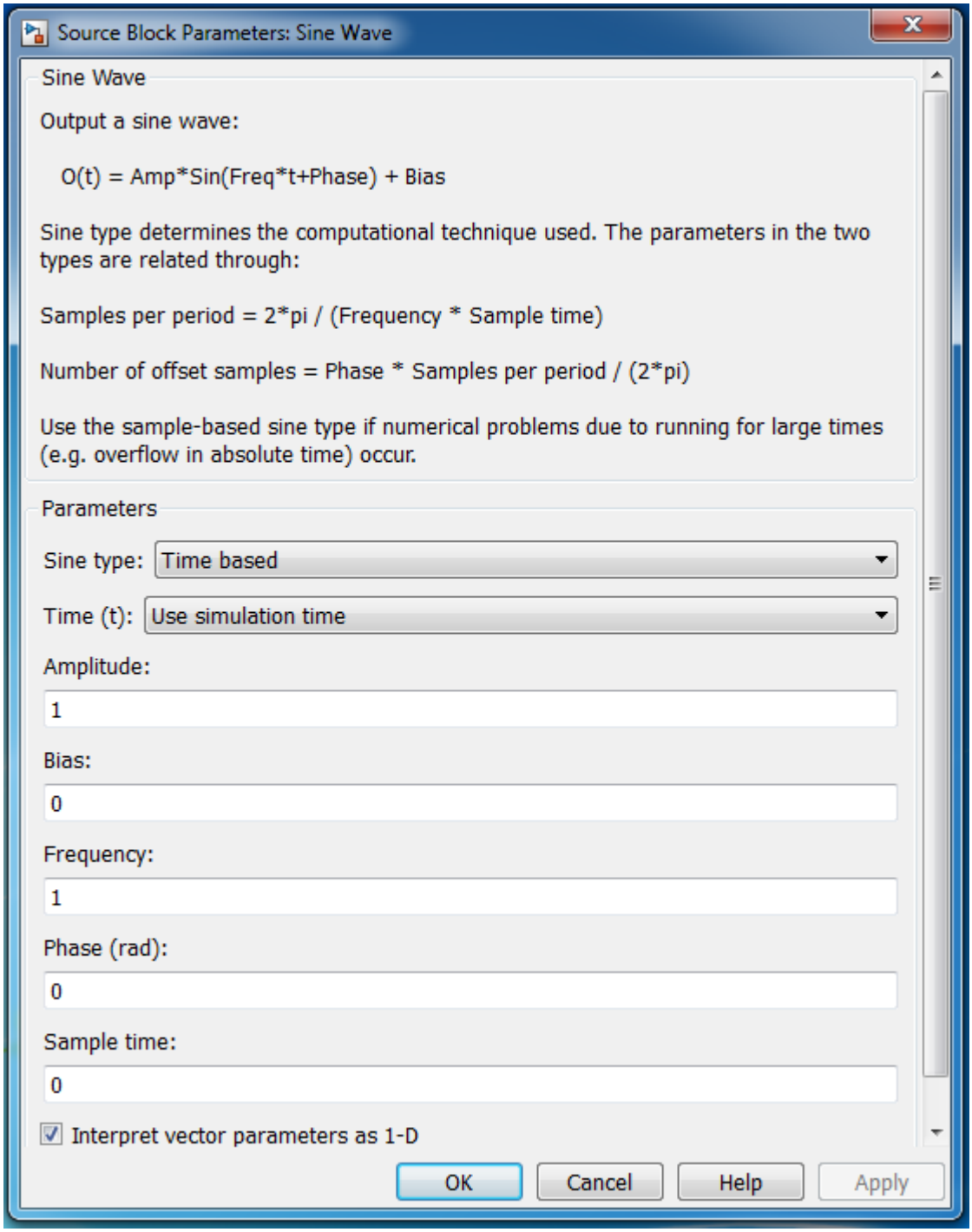
قم بتغيير قيمة Gain إلى 3، ثم لاحظ الخرج.

يمكن أيضاً جعل القيمة تابعة لمتحول، ويجري تعريف المتحول ضمن MATLAB، لنقم بكتابة المتحول G مثلاً ضمن الخانة عوضاً عن القيمة 3، ثم اذهب إلى MATLAB وقم ضمن Command Window أو ضمن ملف Script. أسند لهذا المتحول القيمة 5 مثلاً.



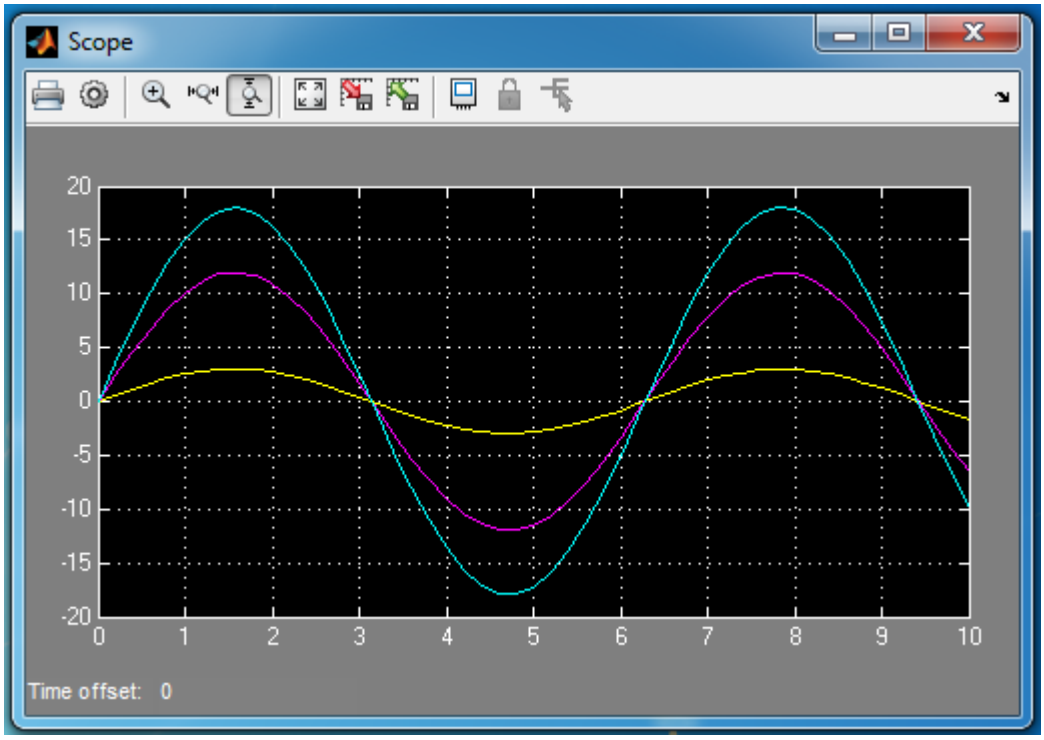
ومن ثم أعد تشغيل المحاكاة ولاحظ الخرج.

تتميز غالبية كتل بيئة Simulink بأنها تدعم الأشعة vectors، مثلاً اضغط على كتلة Sine Wave ستحصل على الشكل التالي:



نلاحظ وجود شرح مفصل عن الكتلة إضافةً للعلاقات التي توصف الخرج وفقاً للمتحولات التي يجري إدخالها ضمن الحقول، سنقوم بتوليد من الإشارات بمطال كختلف وبنفس التردّد، سندخل شعاع المطال التالي ضمن خانة Amplitude:

[6 , 4 , 1] ثم نعيد المحاكاة فنحصل على الشكل التالي:



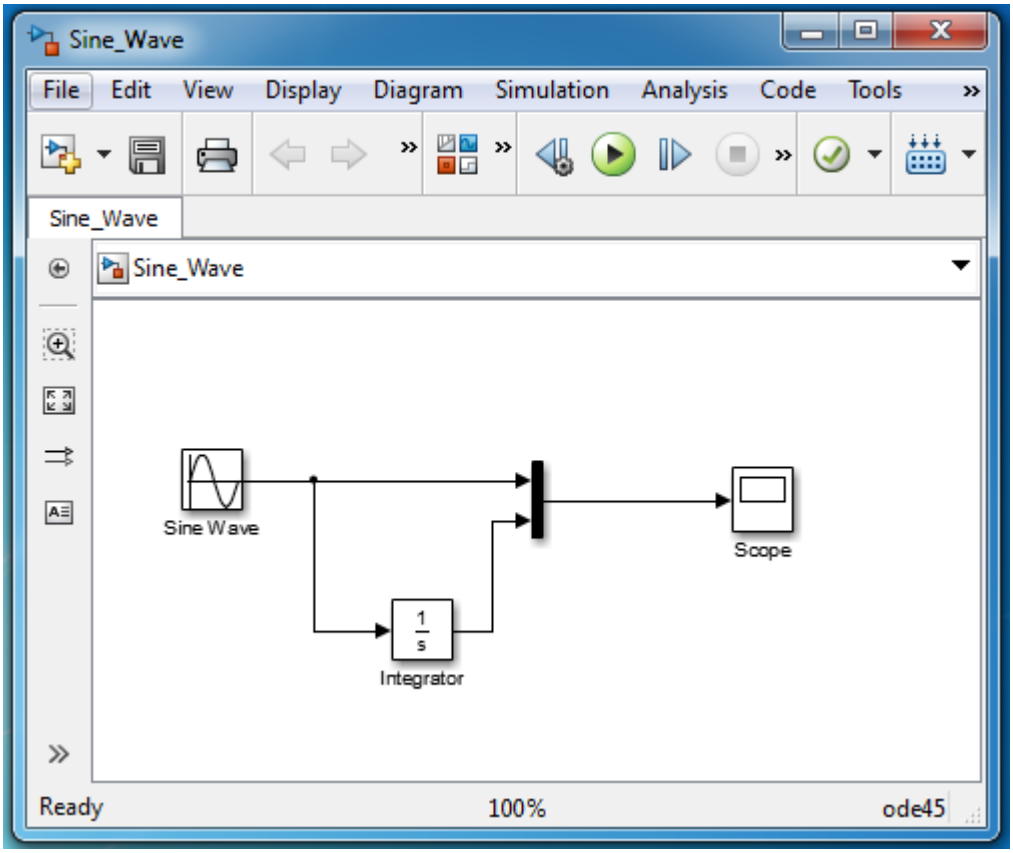
سنضيف إلى المثال السابق تعديلاً بسيطاً، حيث سنقوم أولا بحذف كتلة التضخيم ومن ثم مكاملة الإشارة وعرض الإشارة الأصلية وناتج تكاملها لذلك نحتاج إلى الكتل الإضافية التالية:

- Integrator يستخدم من أجل القيام بعملية المكاملة.
- MUX يستخدم من أجل ضم عدة إشارات، في حالتنا سيقوم بضم الإشارة الأصلية وناتج المكاملة.

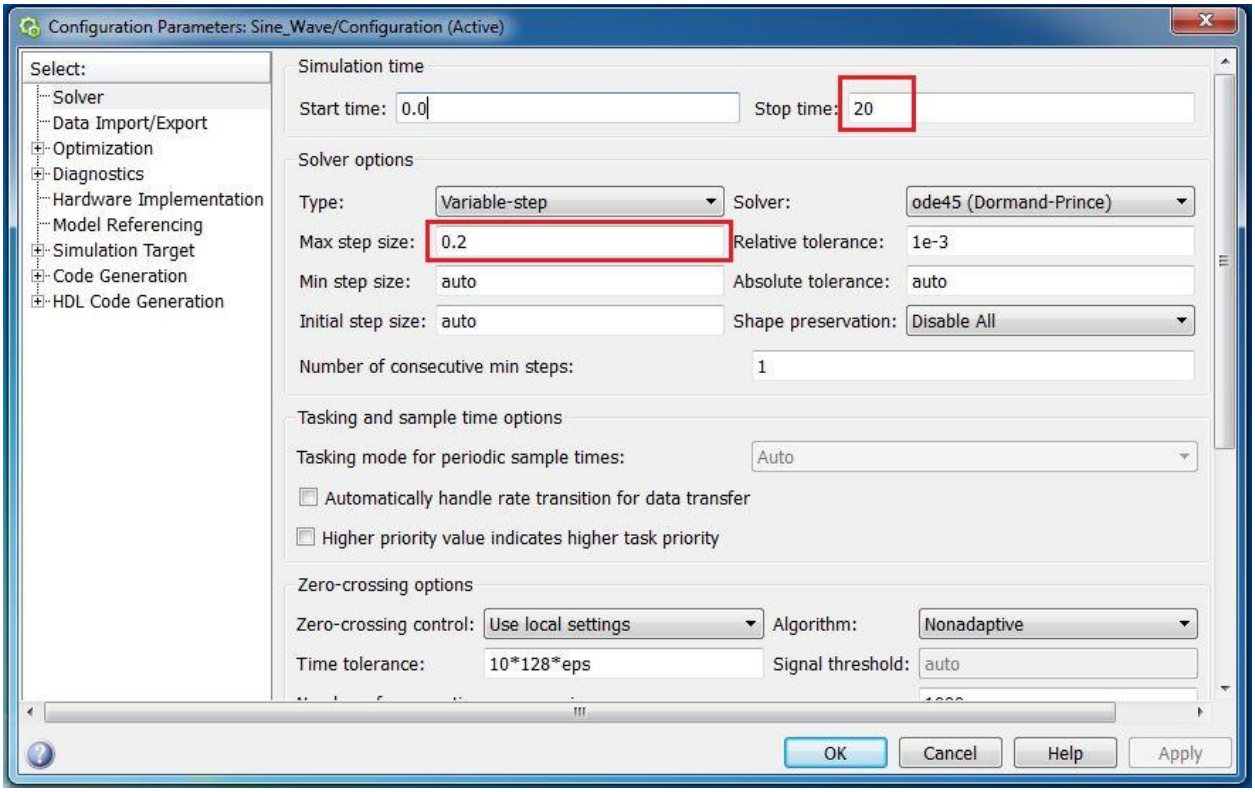
للحصول على MUX و Integrator اذهب إلى Simulink ثم Commonly Used Blocks.

ينصح بترتيب وتنظيم الكتل بشكل منطقي لجعل الخطوط بين الكتل مستقيمة قدر الإمكان.

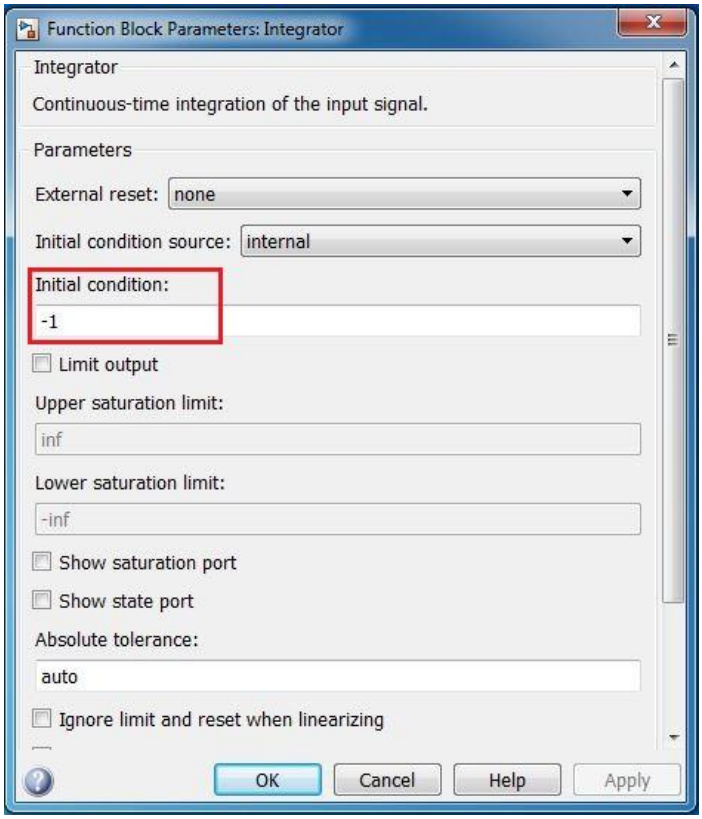
الشكل النهائي للتصميم هو:



قم بتغيير إعدادات المحاكاة لتوافق التالية:

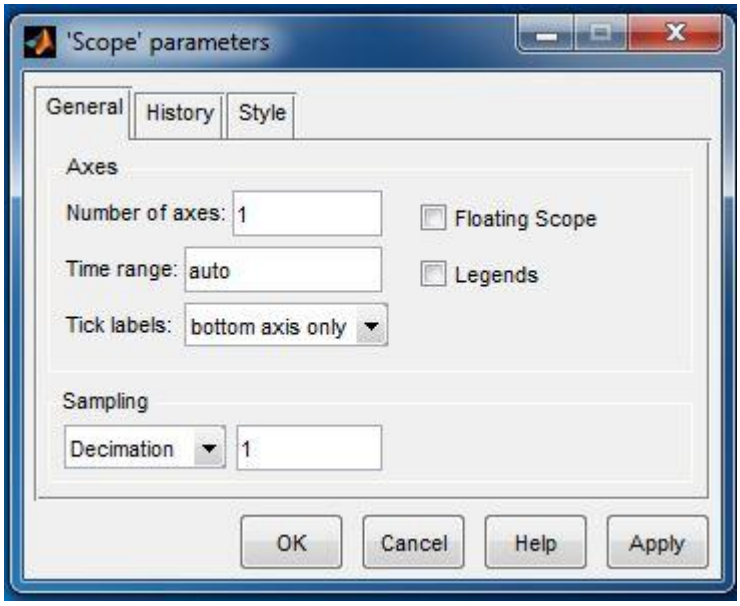


إضافةً لذلك ادخل إلى إعدادات كتلة Integrator وضع القيمة البدائية التالية:



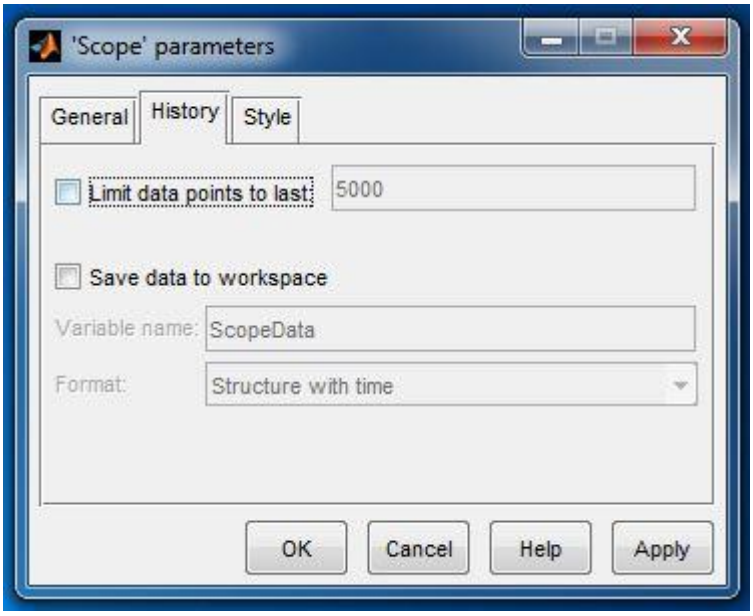
ثم قم بإجراء المحاكاة، اضغط على scope مرتين لمشاهدة المحاكاة، ستلاحظ انه لا يعرض كامل مدة المحاكاة لذلك اضغط ضمن الشريط على الزر الموضح بالشكل

تظهر امامك الواجهة التالية:



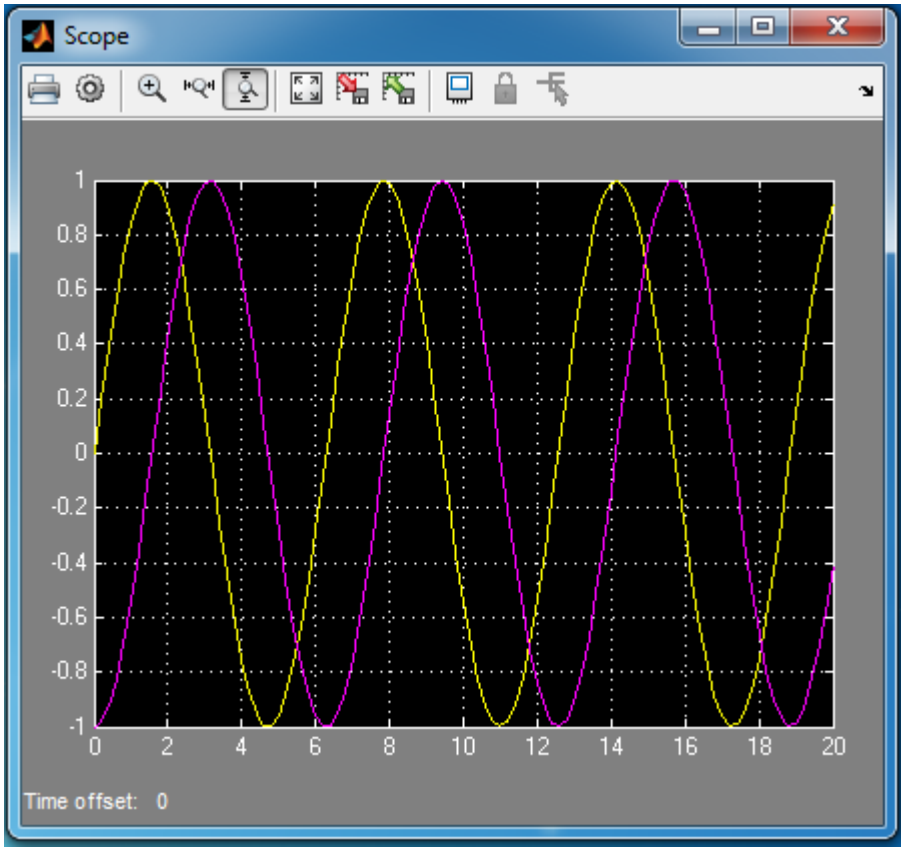
يمكن من خلال الواجهة التالية تحديد عدد المحاور ضمن الشاشة أي عدد الأشكال التي ترغب برسمها .

اذهب إلى History واجعل الخيارات متوافقة مع الشكل التالي، وذلك بهدف عرض جميع نقاط المحاكاة:

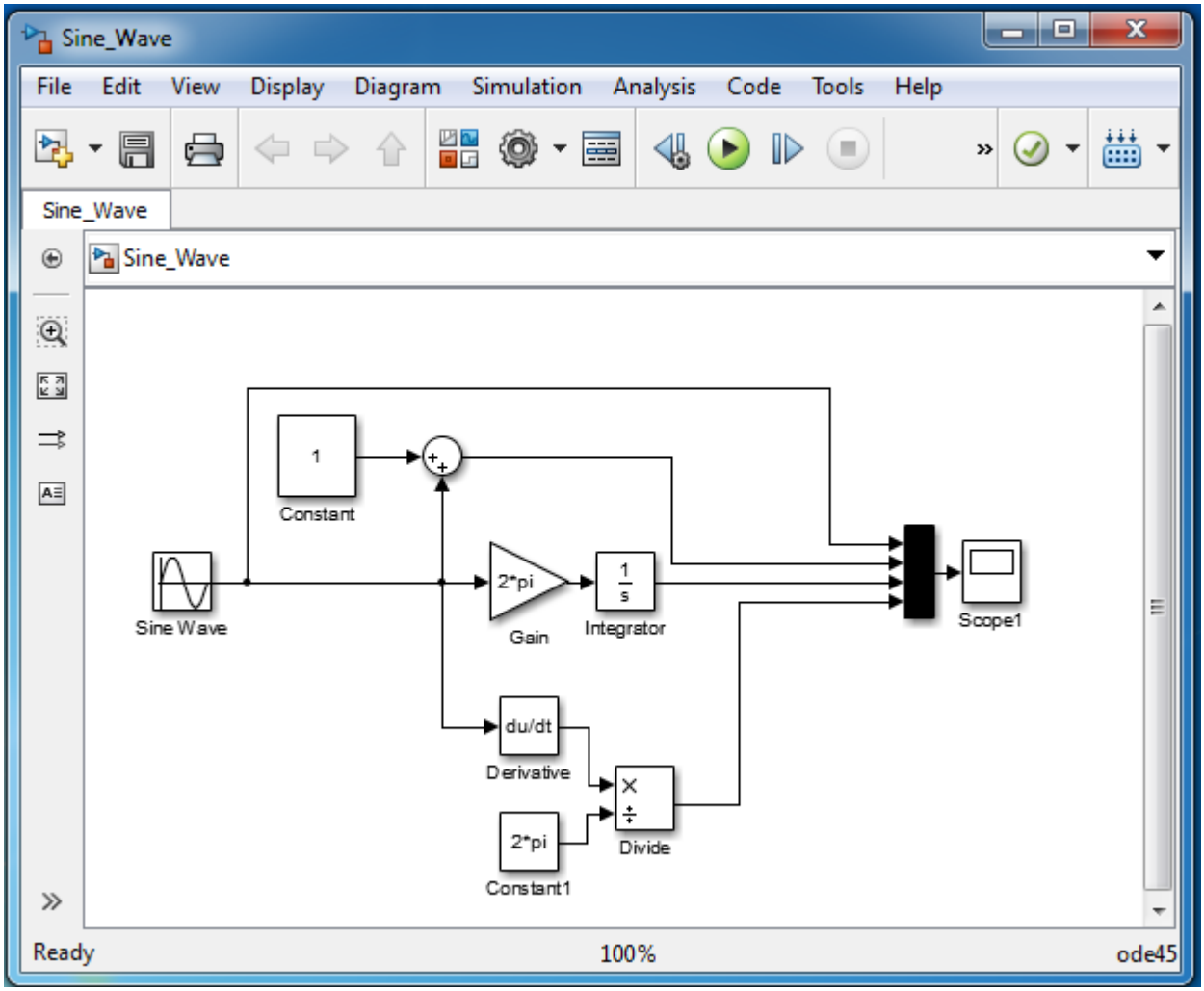


يمكن ضمن النافذة Style التحكم بألوان وشكل وتنسيق الخرج. قم بإجراء تغييرات مثل تغيير لون المحاور والخرج مثلاً.

الخرج الهائي للمحاكاة هو:



سنقوم بإضافة تعديل آخر على التصميم، ليصبح على الشكل التالي:



نحتاج إلى الكتل الإضافية التالية:

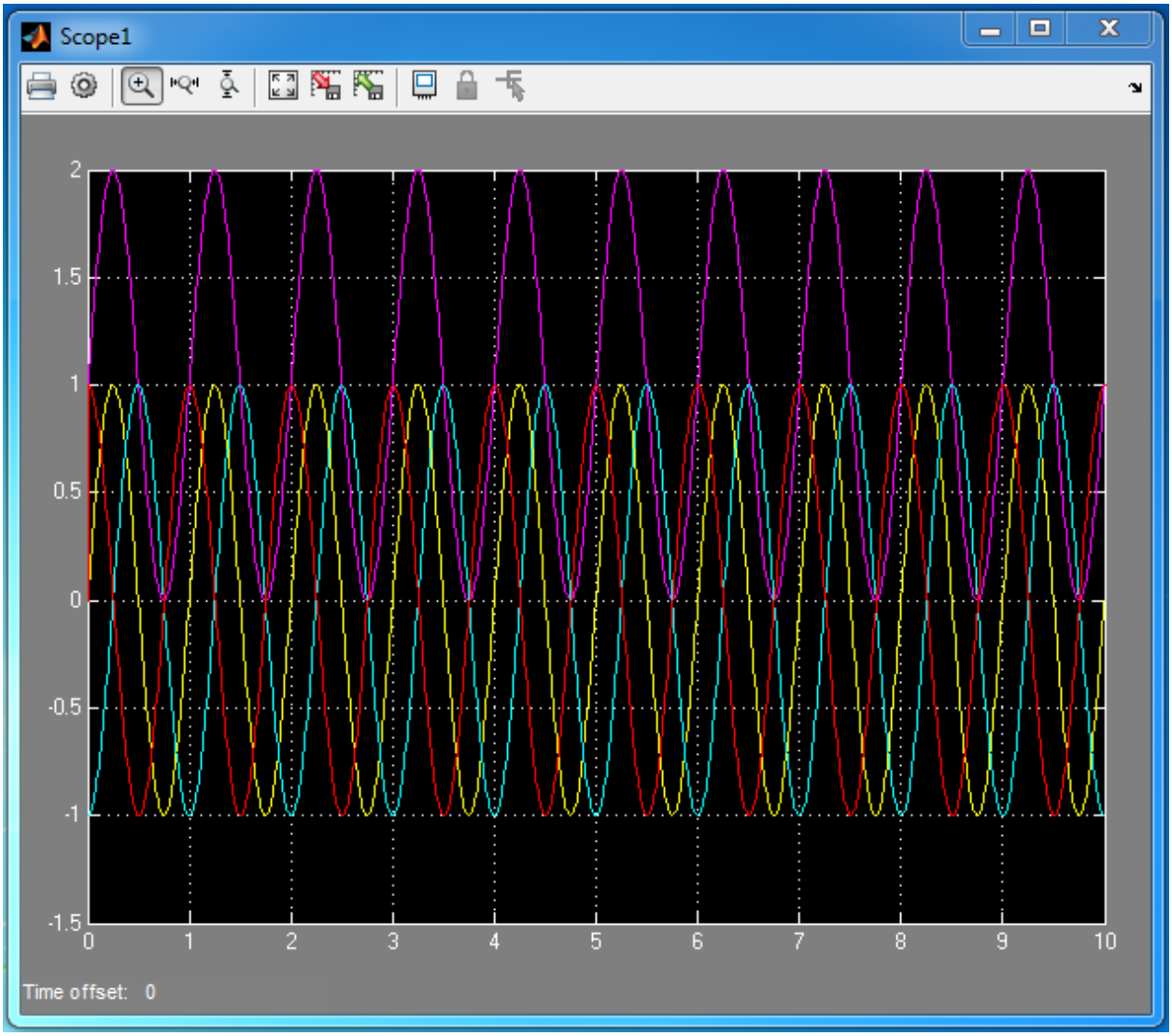
- Constant لإضافة قيمة ثابتة ما.
- Derivative من أجل أخذ المشتق.
- Sum لجمع الإشارتين.
- Divide من أجل التقسيم على قيمة ما.

لنضع المحدّدات التالية ضمن الكتل:

- Sin Wave يجب أخذ تردّد الإشارة Frequency مساوي للقيمة 2*pi.
- Constant الأول القيمة 1 والثاني القيمة 2*pi.
- Gain القيمة للتضخيم هي 2*pi.
- Derivative إبقاء قيمة الثابت c على inf.
- Integrator جعل الشرط البدائي هو 1-.
- MUX يمكن التحكم بعدد مداخله ويجب وضعها 4.

إعدادات المحاكاة هي stop time = 10 و Max step size = 0.01.

الخرج هو على الشكل التالي:



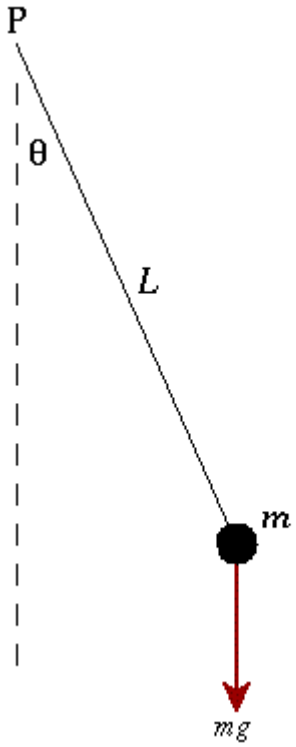
تفسير الخرج: من المخطط الصندوقي للنظام نجد أننا قد أدخلنا إشارة Sin وقمنا بعرضها، أما الإشارة الثانية فقد تم إضافة قيمة ثابتة (تكافئ إضافة جهد DC إلى الإشارة) لذلك نجد الإشارة متمركزة ومتناظرة حول القيمة 1 وهية القيمة الثابتة التي أضفناها. أما الإشارة الثالثة فهي ناتج تكامل إشارة Sin وهي إشارة Cos- ومن أجل الحفاظ على نفس المطال قمنا بالضرب بالقيمة 2*pi أما الإشارة الرابعة فهي مشتق الإشارة Sin وهو Cos ومن أجل الحفاظ على نفس المطال قمنا بالتقسيم على القيمة 2*pi.

تذكر:

$$\frac{d\,(\sin(2\pi x))}{dx} = 2\pi * \cos(2\pi x)$$

مثال:

نرغب في هذا المثال بإجراء محاكاة لحركة نواس بسيط باستخدام MATALB ومن ثم Simulink، وهو موضّح بالشكل التالي:



سنقوم بحل سريع للمسألة لاستنتاج المعادلة التفاضلية لحركة النواس، سنقوم ببناء حدود المعادلة التفاضلية باستخدام الكتل blocks المعرفة مسبقاً ضمن Simulink.

الحل:

نعلم أن مجموع العزوم = عزم العطالة مضروباً بالتسارع الزاوي

العزم يساوي القوة مضروبة بالذراع.

عزم العطالة يساوي الكتلة بمربع البعد وذلك من أجل نقطة مادية.

يوجد قوتين تؤثران على الجسم الثقل، وتوتر الخيط، حيث أن قوة توتر الخيط معدومة لأنها موازية لمحور الدوران.

بناءً على ما سبق نكتب

$$-m.g.\ell.\sin(\theta) = m.\ell^2.\ddot{\theta}$$

$$\ddot{\theta} + \frac{g}{\ell}.\sin(\theta) = 0 \qquad (1)$$

يمكن أخذ تقريب من أجل زوايا صغيرة أي $\theta \simeq 0$ وعنها يكون $\sin(\theta) \simeq \theta$، عندها تصبح المعادلة التفاضلية على الشكل:

$$\ddot{\theta} + \frac{g}{\ell}.\theta = 0 \qquad (2)$$

ويكون الدور T معطى بالعلاقة $T = 2\pi.\sqrt{\frac{\ell}{g}}$.

سنستخدم المعادلة التفاضلية (1) للحل أولاً باستخدام MATLAB ثم سنستخدم Simulink.

الحل سيكون على الشكل التالي:

$$\ddot{\theta} + \frac{g}{\ell}.\sin(\theta) = 0$$

$$\ddot{\theta} = -\frac{g}{\ell}.\sin(\theta)$$

لنفترض أن $\dot{\theta} = \gamma$ عندها يكون

$$\gamma = -\frac{g}{\ell}.\sin(\theta)$$

ومنه التابع سيحتوي على عنصرين أي الدخل هو شعاع على الشكل:

$$\hat{x} = \begin{bmatrix}\theta \\ \gamma\end{bmatrix}$$

$$\frac{d\hat{x}}{dt} = \begin{bmatrix}\dot{\theta} \\ \dot{\gamma}\end{bmatrix}$$

نقوم أولاً بإنشاء ملف تابع function يحتوي على الرماز التالي، لا تنسى أن يكون التابع ضمن نفس مسار العمل عند استعماله ولا تنسى أن يكون اسم الملف المحفوظ به التابع هو نفس اسم التابع.

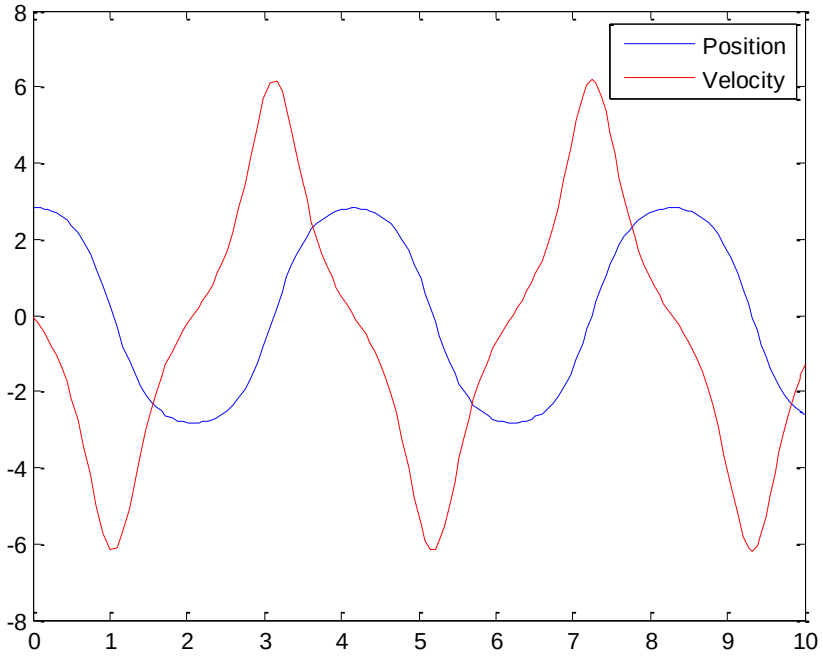
```
function dxdt = pendulum_eq(t,x )
% Pendulum Equations
L =1;
theta = x(1);
gamma = x(2);

dtheta = gamma;
dgamma= - (9.8/L)*sin(theta);

dxdt = zeros(2,1);
dxdt(1)=dtheta;
dxdt(2)=dgamma;
end
```

قم بإنشاء ملف script واكتبه ضمنه الرماز التالي:

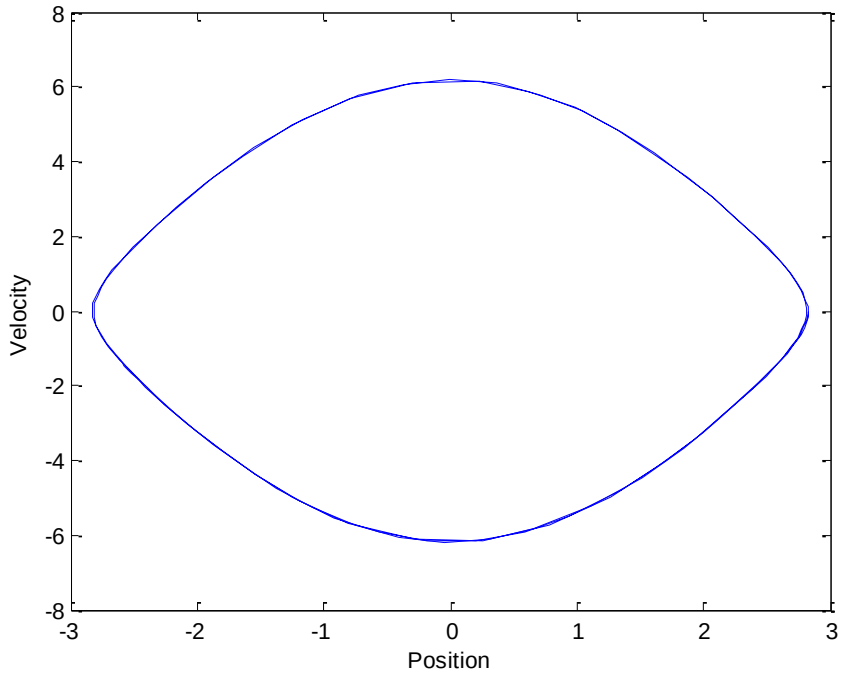
```
%% ODE pendulum
% assume pendulum is almost horizontal
```

```
[t,x]=ode45('pendulum_eq',[0 10],[0.9*pi 0]);
plot(t,x(:,1));
hold on;
plot(t,x(:,2),'r');
legend('Position','Velocity')
```

الخرج المنحني الأزرق هو الموضع بمعنى الزاوية مقدراً بـ rad، والمنحني الأحمر هو السرعة مقدرةً ب m/s.

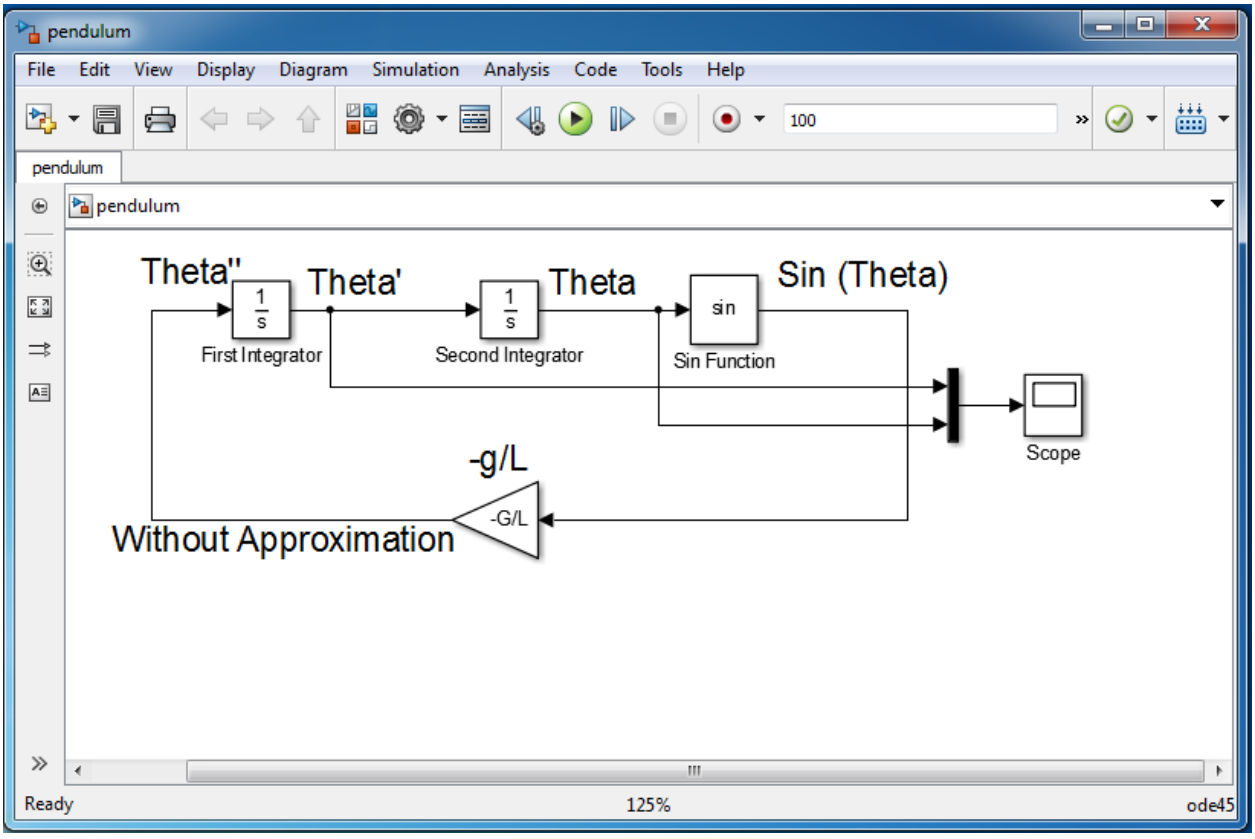


من الممكن أيضاً رسم السرعة بدلالة الموضع، نلاحظ أن السرعة معدومة عندما تكون قيمة theta أعظمية، وعندما تكون theta =0 تكون السرعة أعظيمة.



سنقوم الآن بتمثيل المعادلة التفاضلية (1) باستخدام Simulink، المخطط النهائي للدارة هو على الشكل التالي:

يجب البحث عن العناصر وتنفيذ النموذج التالي ضمن ملف جديد للمحاكاة، يجري حفظه باسم pendulum.



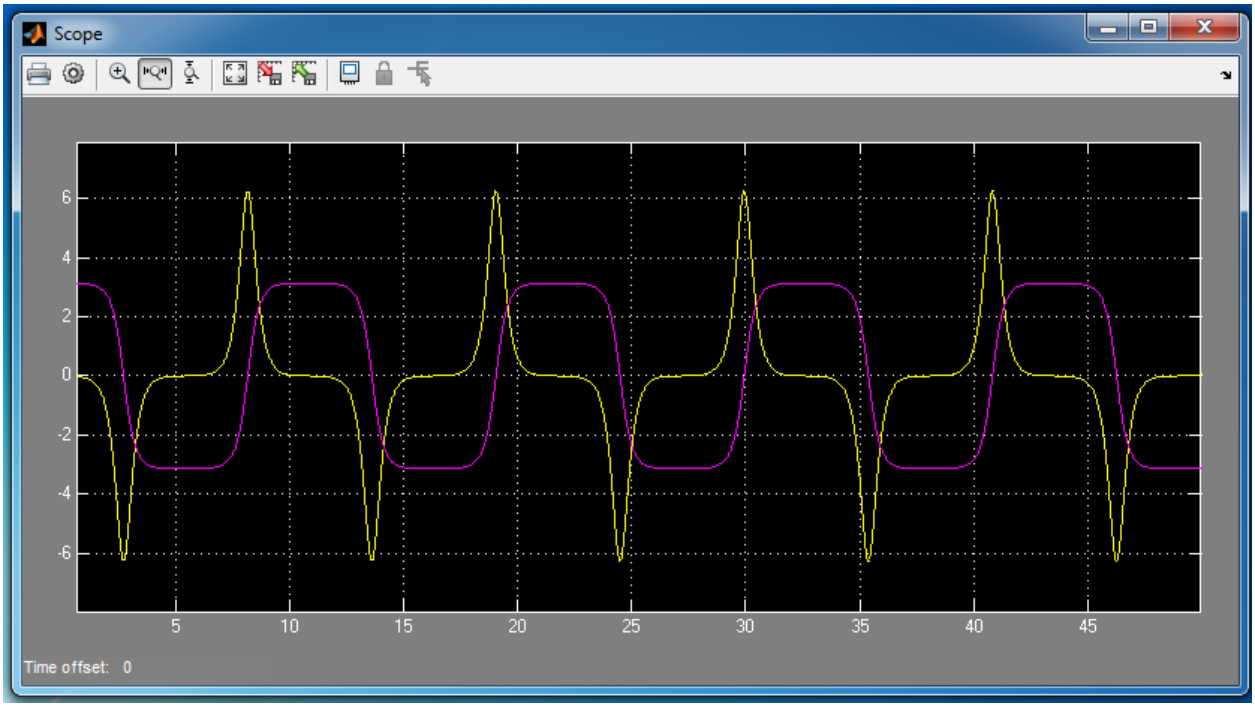
ملاحظة: لإدراج كتابة يمكن الضغط مرتين بالزر اليساري للفأرة على أي مكان ضمن النافذة ومن ثم الكتابة.

ملاحظة:الربح ضمن المضخم هو G/L- إضافةً للقيمة الابتدائية ضمن المكامل الأول هي theta_der0 وضمن المكامل الثاني theta0 يجب تعريف المتحولات ضمن Command Window أو ضمن ملف script.

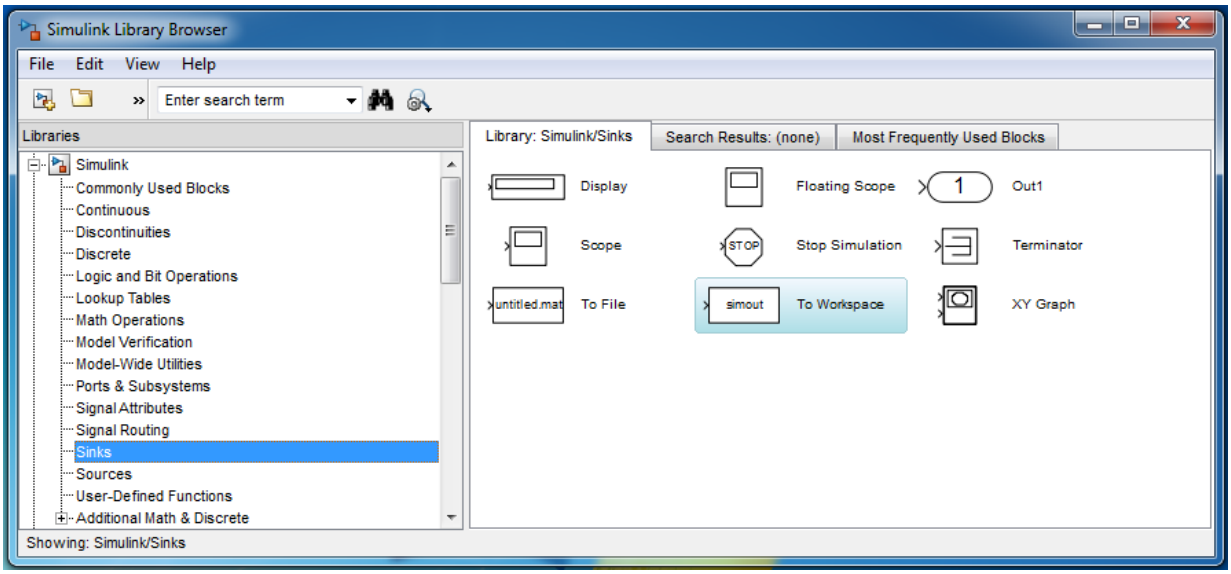
```
%% Simulink Example
% Simple Pendulum
% Here we define variable and initial condition
G=9.8; % Gravity Acceleration Constant
L= 1;% Length of Pundulum = 1 m
theta_der0= 0;% Initial Value of speed = 0
theta0=3.14;% Initial Value of position
```

ملاحظة: قم بتغيير إعدادات المحاكاة واجعل stop time = 100 و Max step size = 0.001.

الخرج على الشكل وهو يعبر عن السرعة إضافةً إلى الموضع، قم بإجراء مقانة مع الخرج النتج عن الحل باستخدام MATLAB.

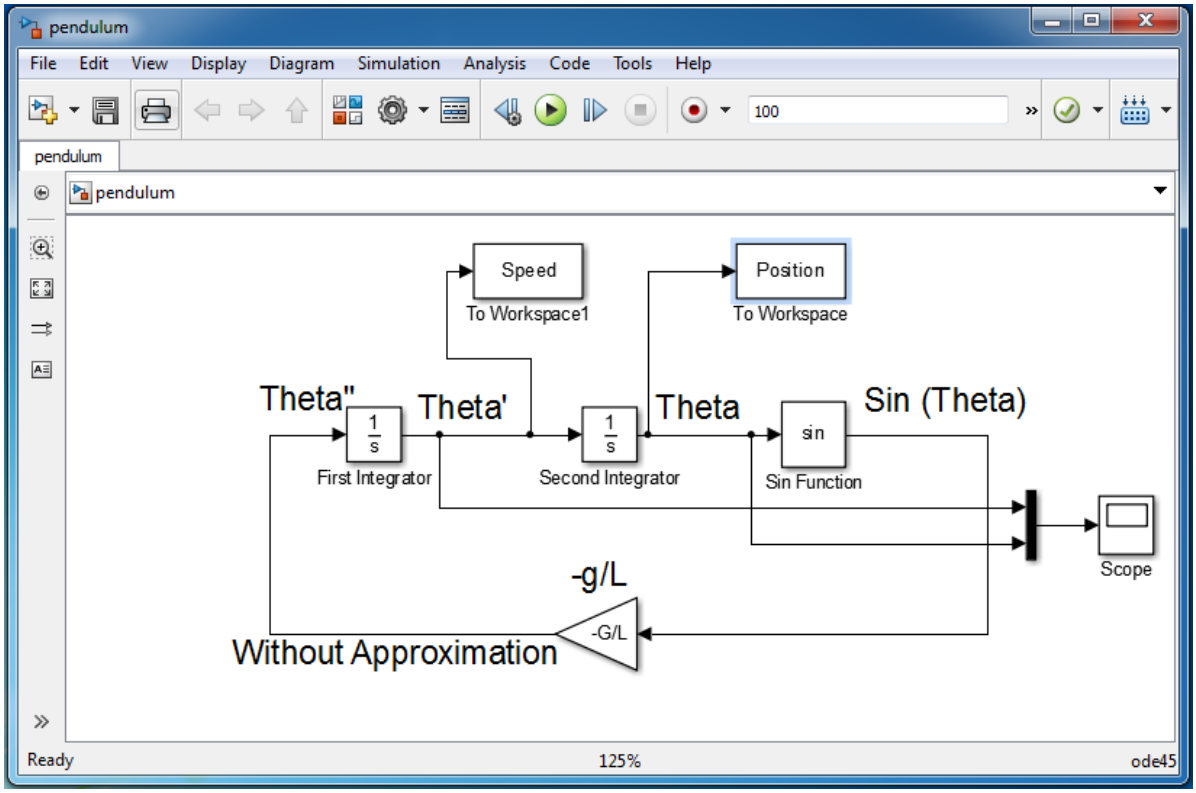


يوجد كتل عديدة مهمة أيضاً نذكر منها to workspace و to file.

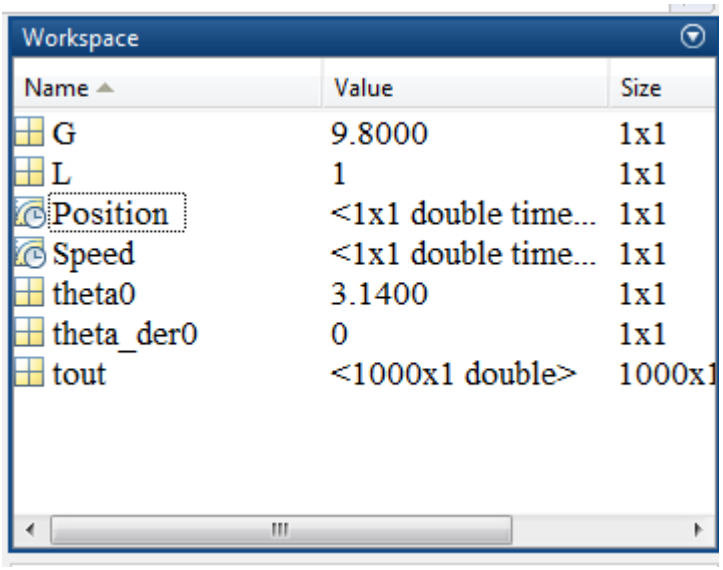


اقرأ ضمن help عمل كل منهما، سنستخدم to workspace من أجل الحصول على نتائج المحاكاة ومن ثم نقوم برسمها ضمن MATLAB، حيث يعتبر الحصول على النتائج أمراً مهماً بهدف إجراء معالجة لاحقة عليها سنكتفي هنا بالرسم فقط.

قم بوصل هذه الكتل للحصول على شعاع السرعة والموضع كما في الشكل، يوجد متحول ضمن كل كتلة سمّه Speed في الكتلة الأولى و Position في الكتلة الثانية.



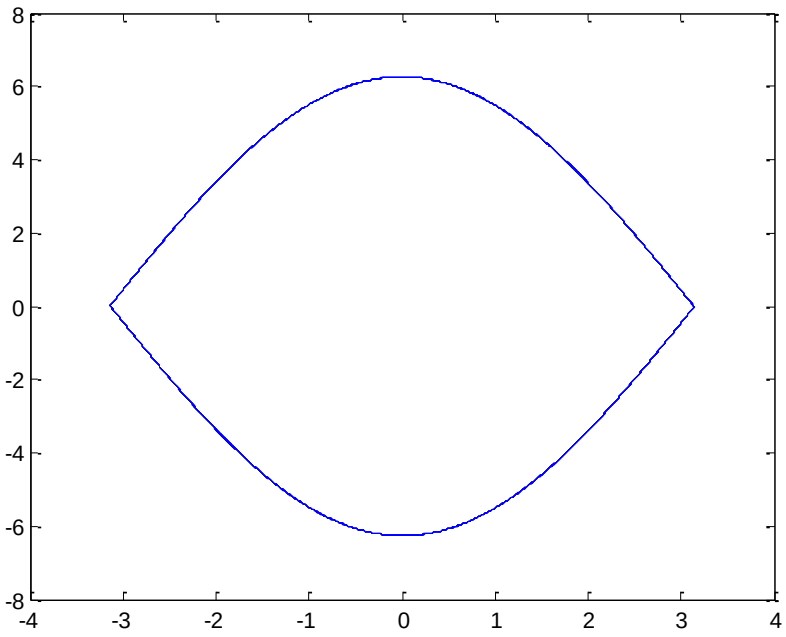
عند تشغيل المحاكاة نلاحظ تعريف متحولين Position و Speed ضمن Workspace وكل منهما عبارة عن Timeseries وهي بنية معطيات تحتوي على حقلين الأول شعاع الزمن والثاني حقل المعطيات للوصول إلى المعطيات مثلاً نستخدم Positon.Data، يوجد خيارات أخرى للحفظ سنتسخدمها في الأمثلة اللاحقة.



لنقم الآن برسم السرعة بدلالة الموضع كما هية الحالة عند تنفيذ التمرين باستخدام MATLAB.نكتب مايلي:

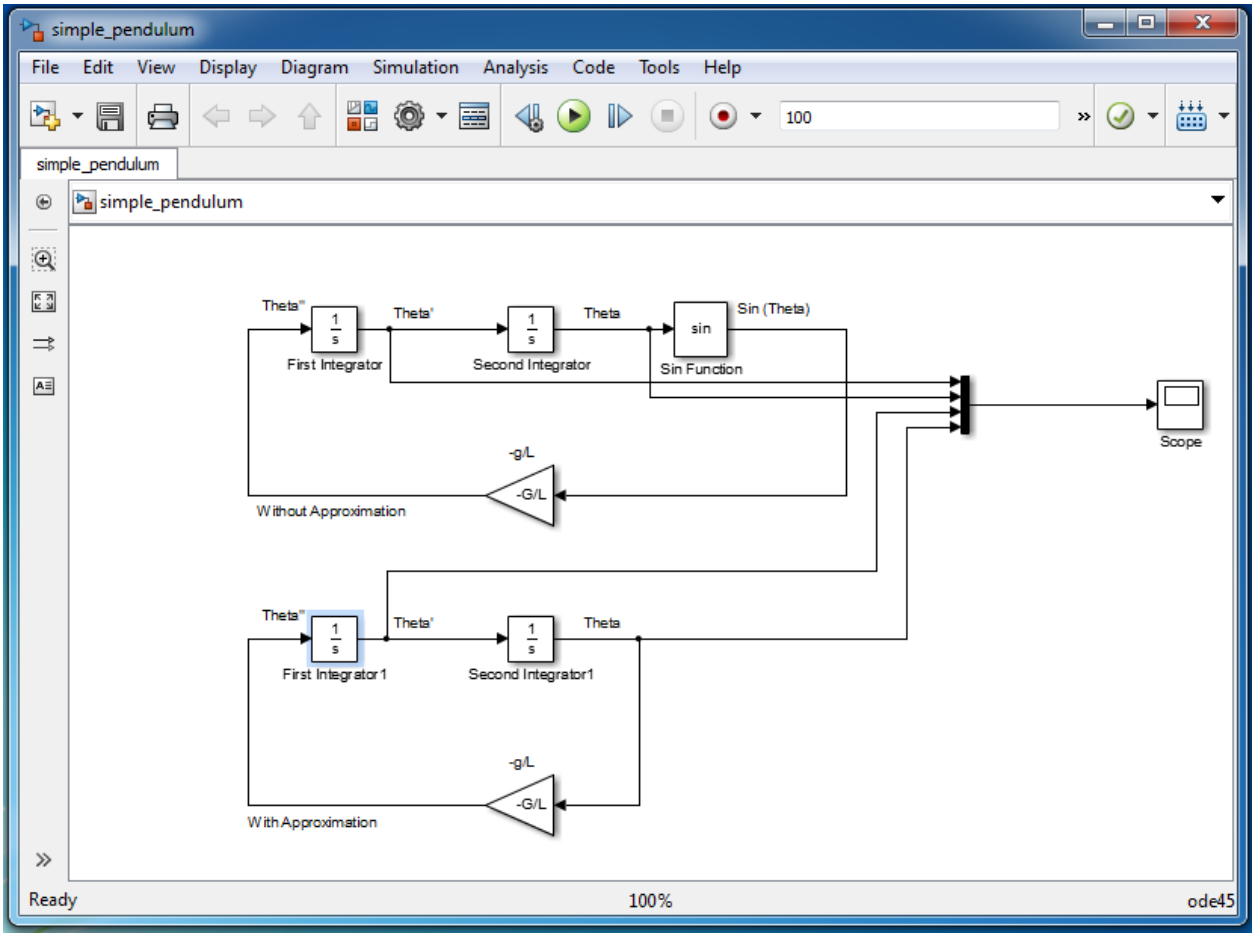
```
plot(Position.Data,Speed.Data)
```

نحصل على الشكل التالي:



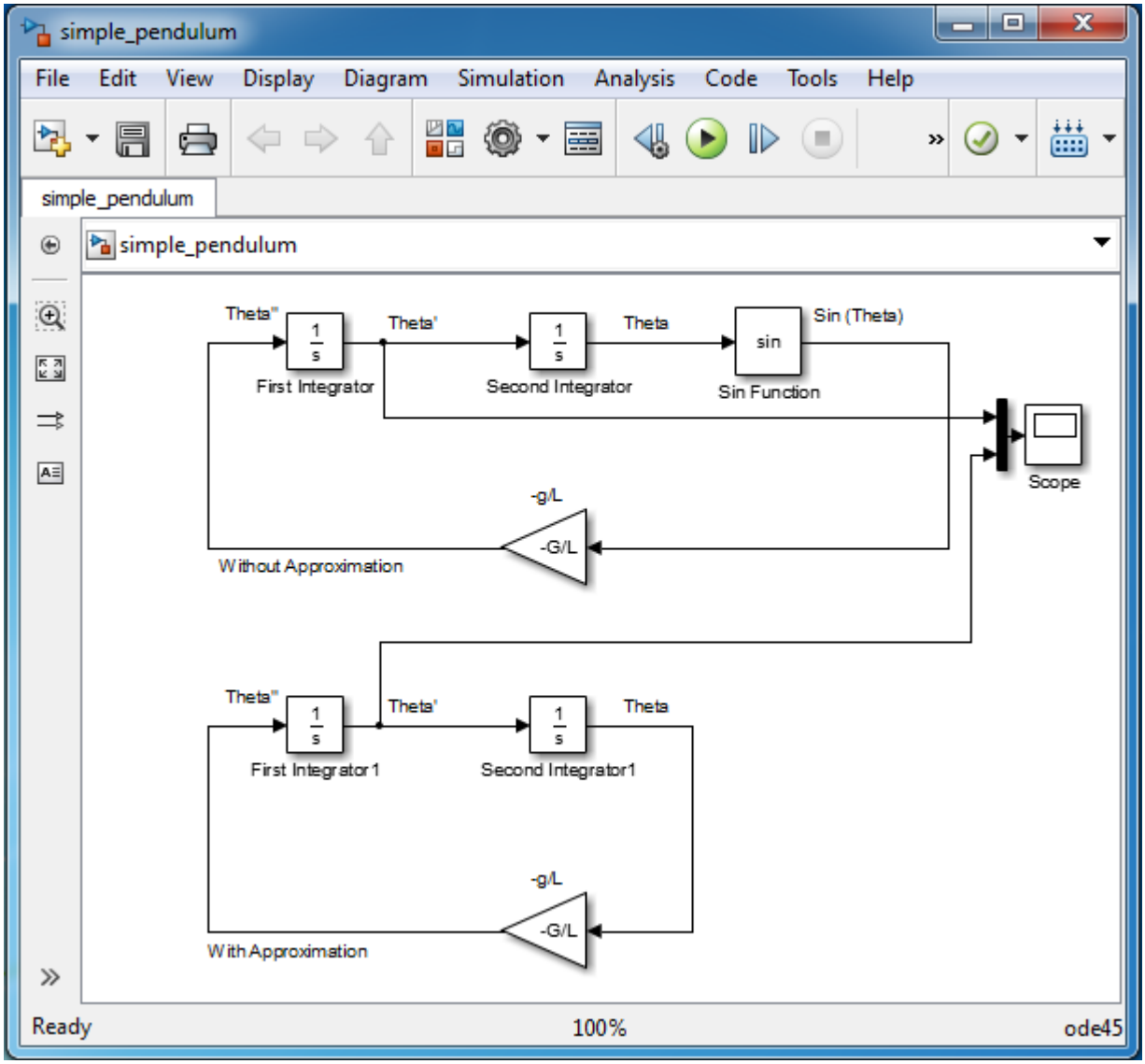
الشكل السابق يشابه الشكل الذي حصلنا عليه باستخدام MATLAB، وبهذا نجد كيفية الوصول إلى المعطيات ويمكن عندها المقارنة بين النتائج مثلاً.

سنقوم الآن بالانتقال إلى المعادلة (2) التي توصف الحركة ولكن مع أخذ تقريب وسنقوم ببناء نموذج محاكاة مع النموذج السابق من أجل المقارنة. الشكل النهائي للنموذجين هو كالتالي:



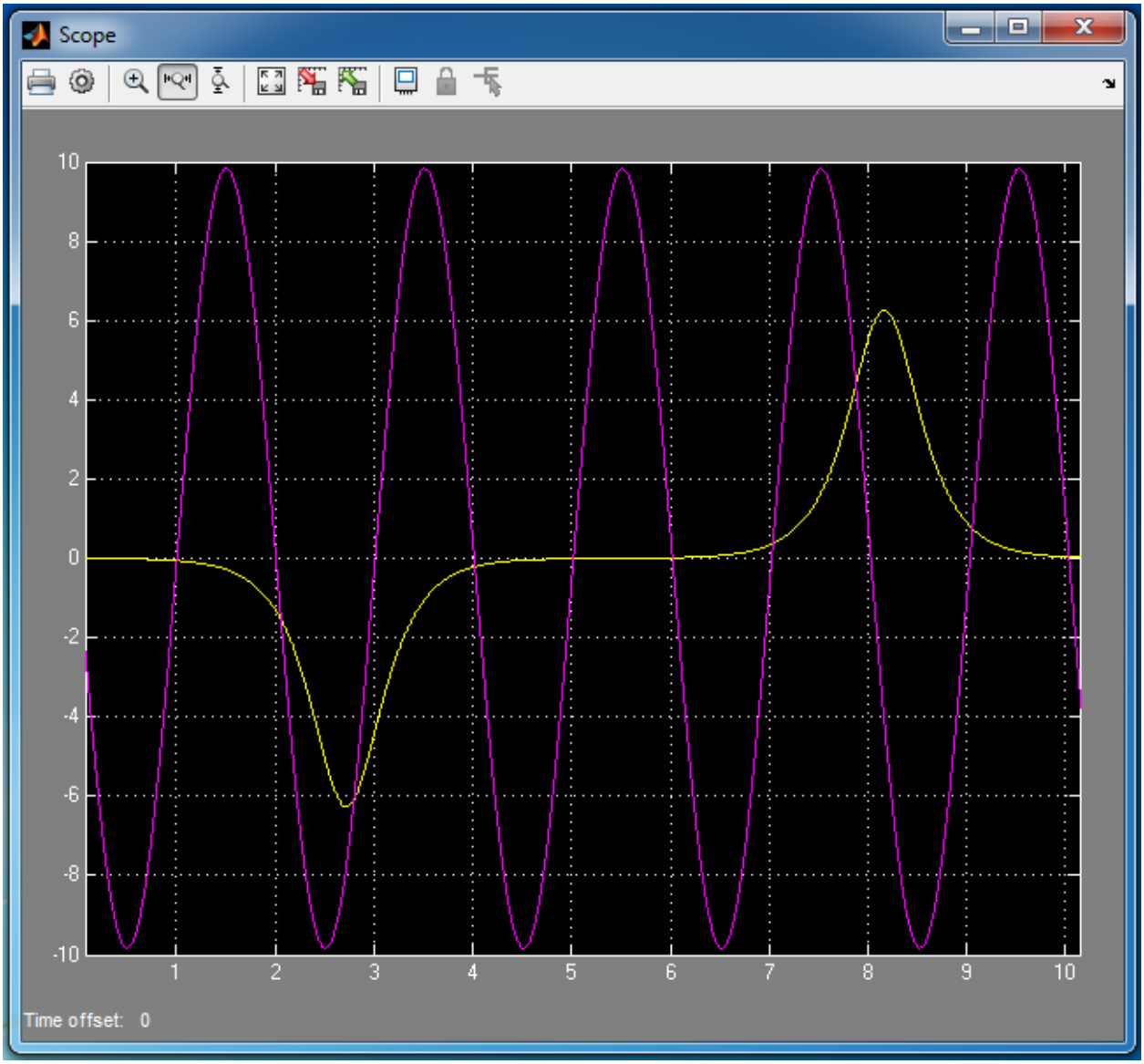
ملاحظة: يجب تعريف نفس القيم البدائية لذلك ينصح بنسخ الشكل الشابق وحذف كتلة sin فقط.

سنقوم أولاً بعرض السرعة فقط يمكن حذف الخطوط التي تمثل الموضع وجعل النموذج يشابه الشكل التالي:

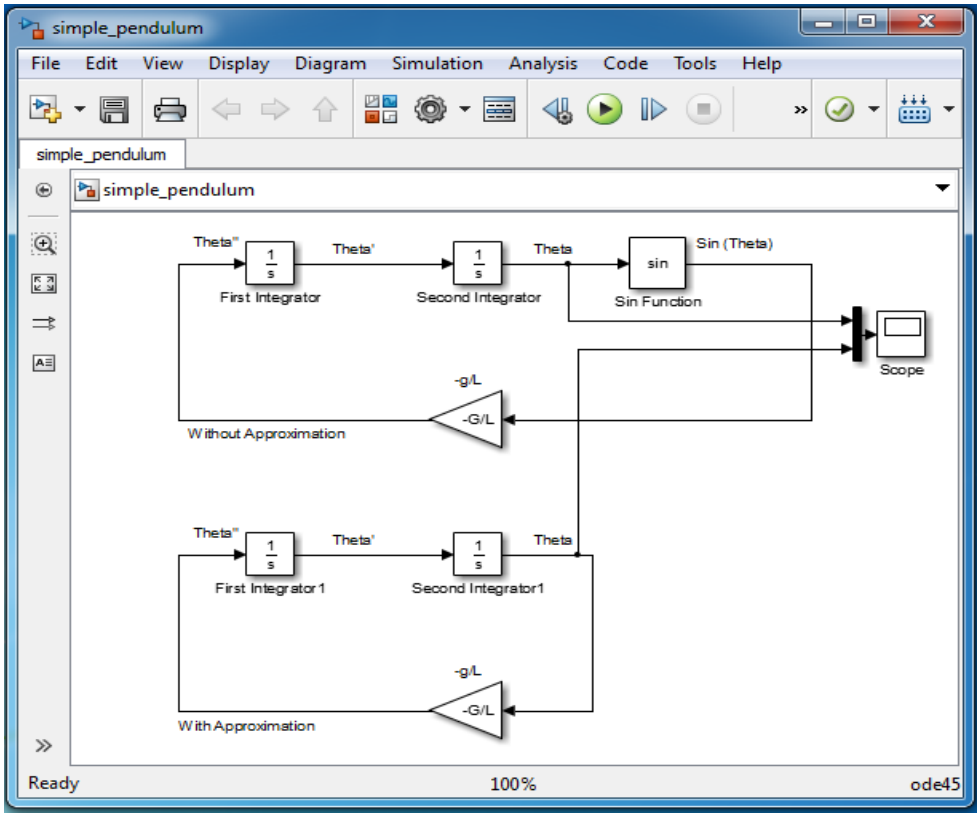


الخرج هو:

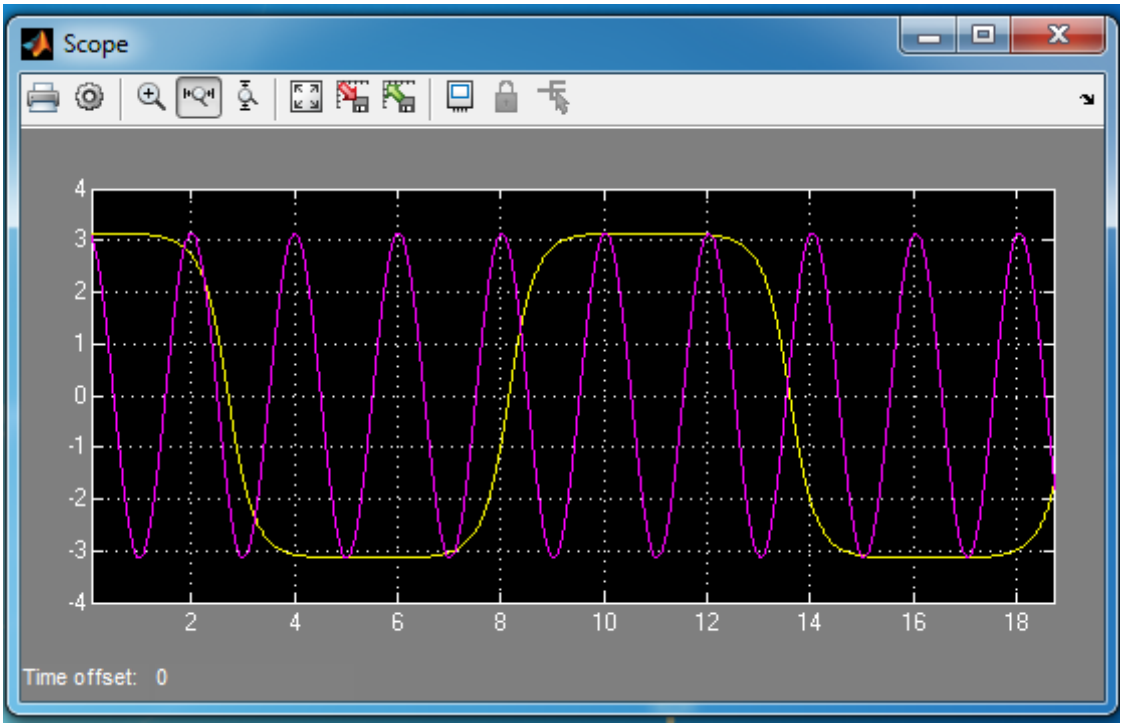
نلاحظ ان السرعة في الحالة الثانية هي تابع جيبي، وهذا متوقع (انظر المعادلة رقم (2)) كما أننا اقتطعنا نافذة زمنية لعرض النتائج ضمنها.



سنقوم الآن بعرض الموضع فقط يمكن حذف الخطوط التي تمثل السرعة وجعل النموذج يشابه الشكل التالي:



الخرج: كما هو متوقع الحالة الثانية للموضع الحل هو تابع جيبي، كما أننا قمنا بعرض النتائج على نافذة زمنية.



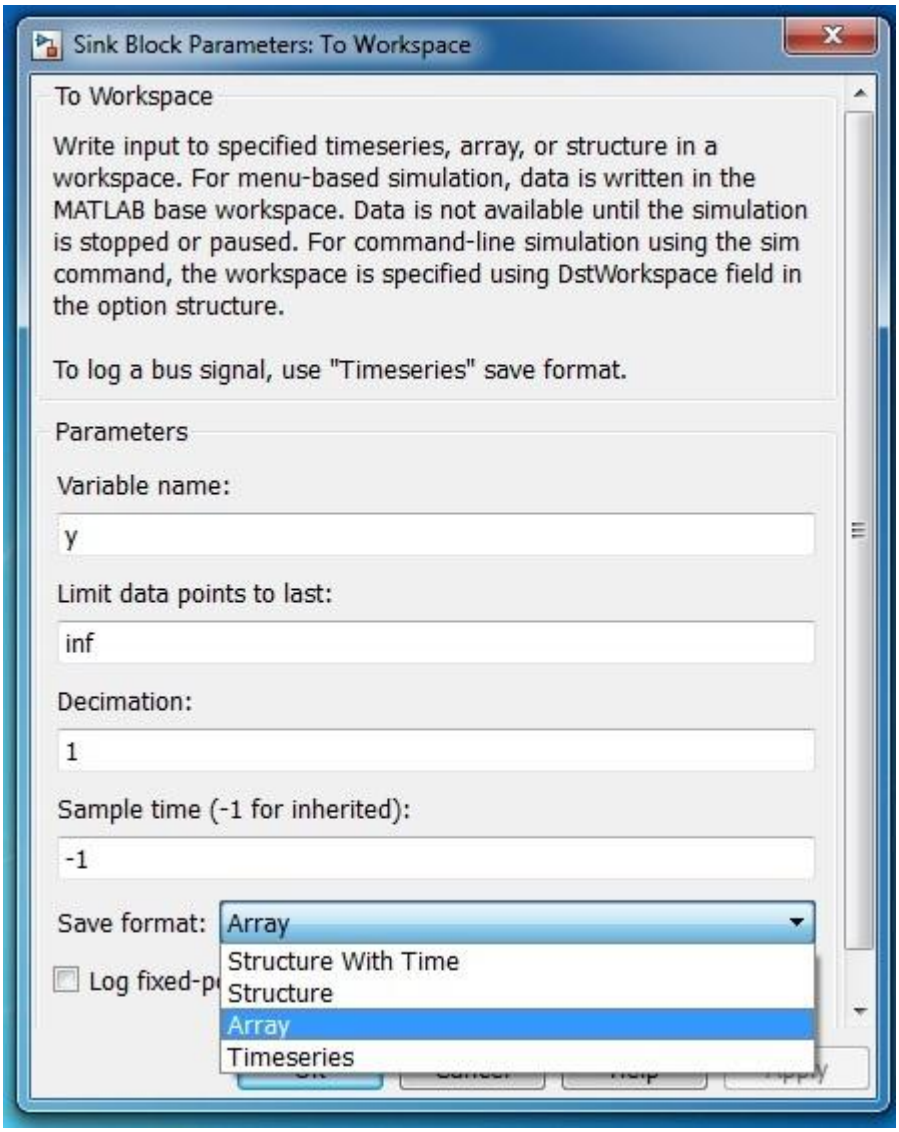
مثال: سنقوم في هذا المثال بتشكيل معادلة تفاضلية معبرة عن نظام ما وإيجاد الحل لها باستخدام Simulink.

المعادلة هي:

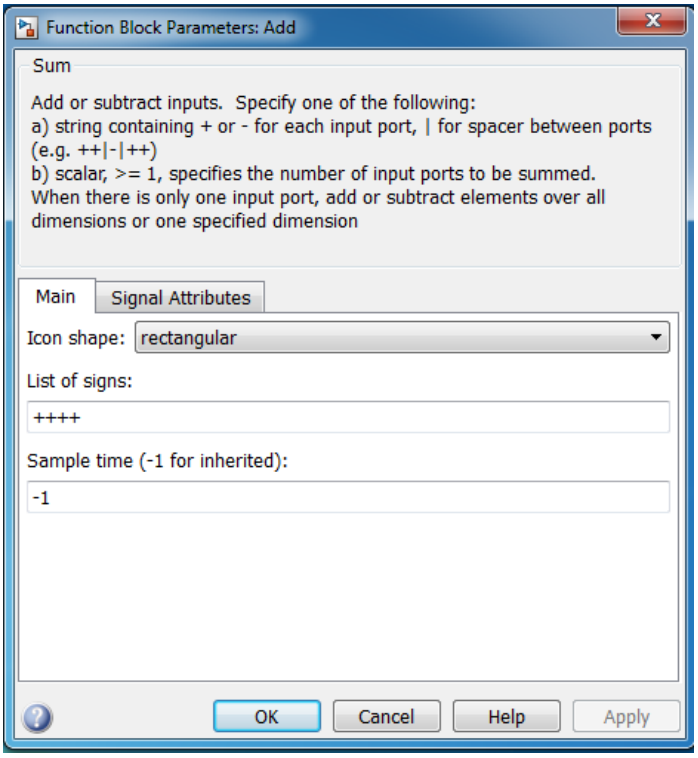
$$\dddot{y} + 3\,\ddot{y} + 2\dot{y} + y = u$$

الشروط البدائية هي: $y(0) = 0; \dot{y}(0) = 0;\ \ddot{y}(0) = 0;$ و u هو إشارة مربعة.

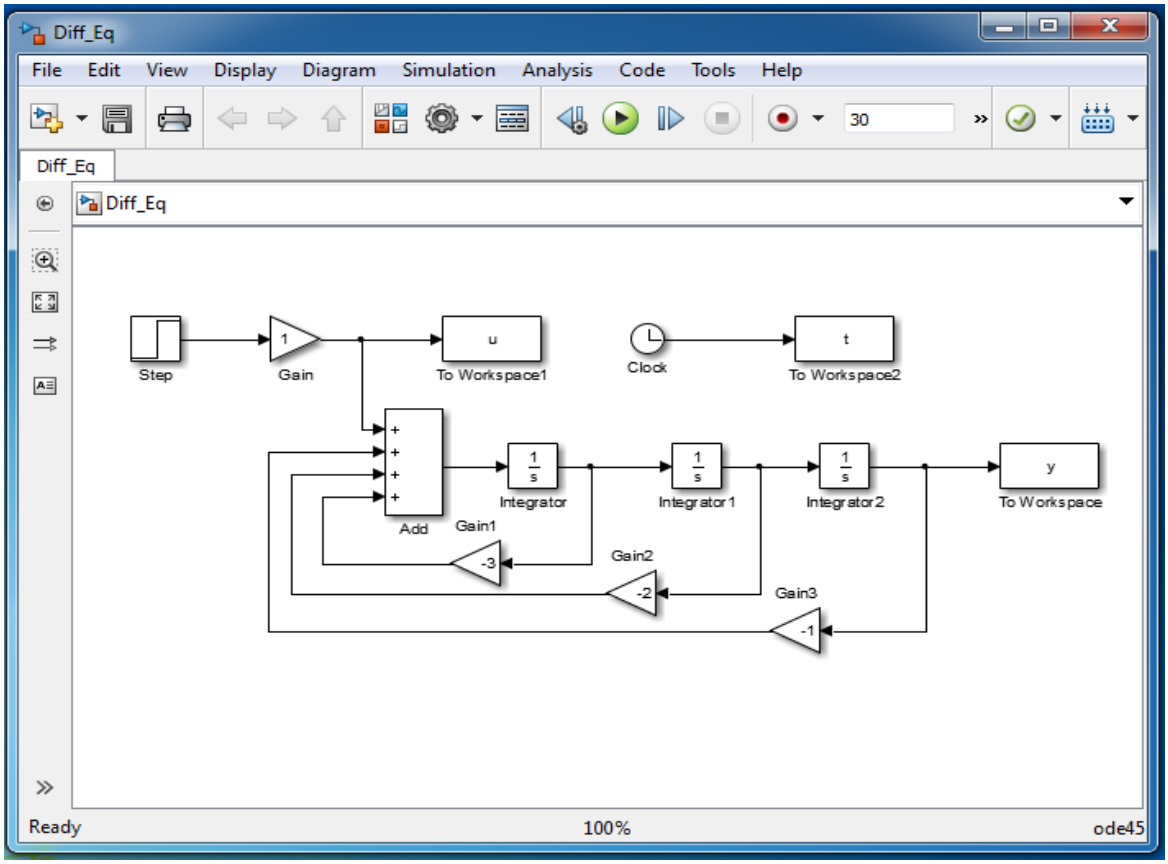
ملاحظة: عند وضع كتلة to workspace وعلى خلاف المثال السابق سنستخدم خيار لحفظ المتحول وهو array.



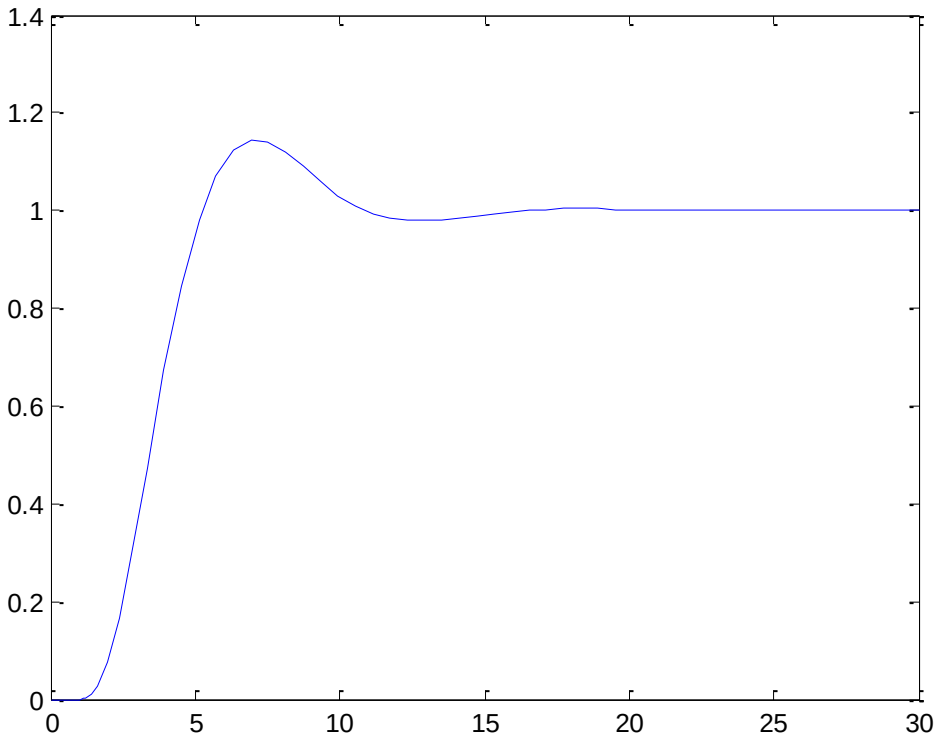
ملاحظة: من أجل توسيع عنصر add إلى 4 مداخل قم بفتح العنصر وضع داخله أربع إشارات + للدلالة على أربع مداخل والعملية المرغوبة هي الجمع كما في الشكل:



الشكل النهائي للنظام هو:



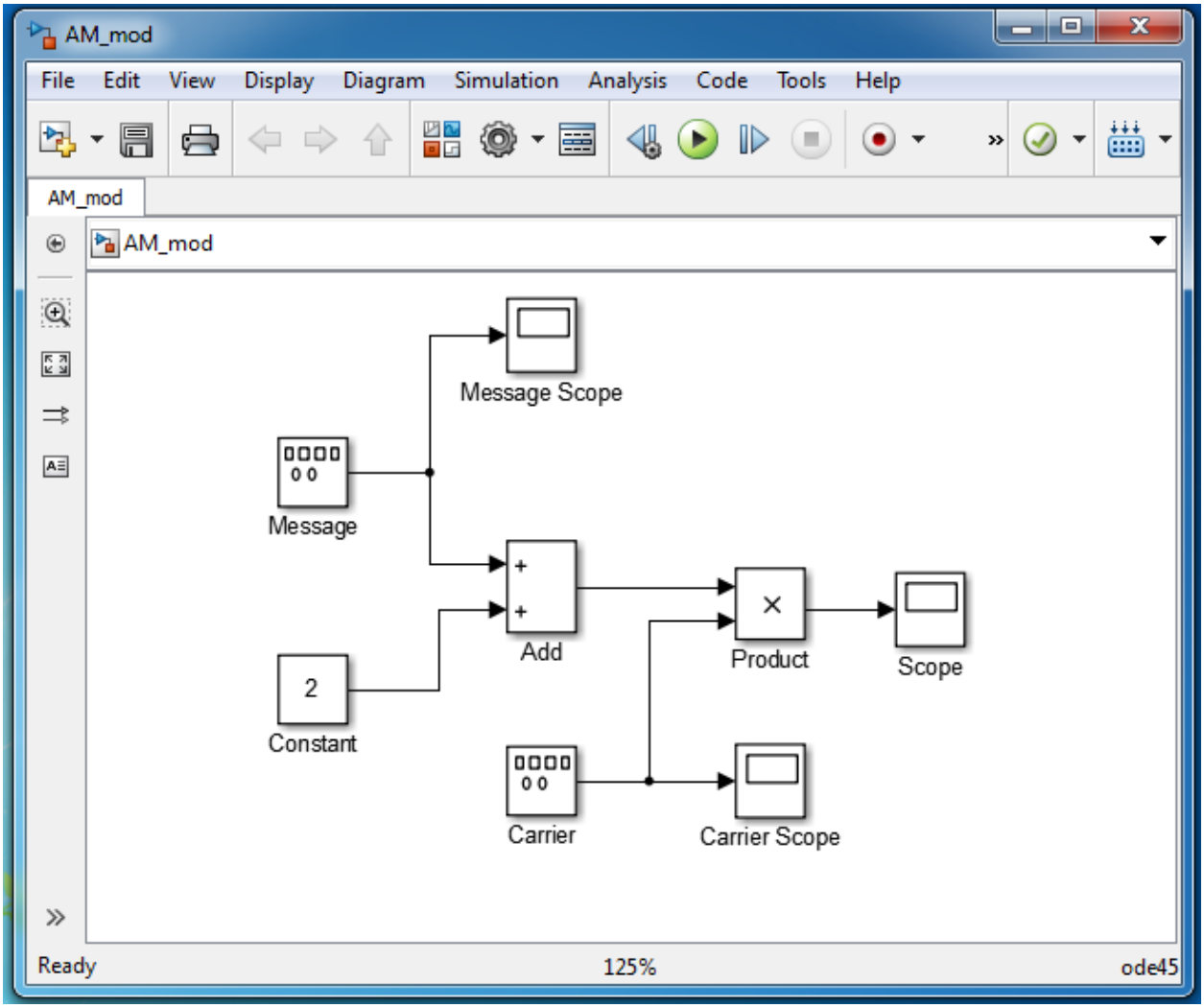
قم برسم المتحول y بدلالة الزمن t لنحصل على الشكل التالي:



مثال: سنقوم في هذا المثال بإنشاء نظام يحاكي التعديل المطالي في أبسط أشكاله، يعتمد التعديل المطالي على تغيير في مطال الإشارة ليكون موافقاً لإشارة المعلومات.

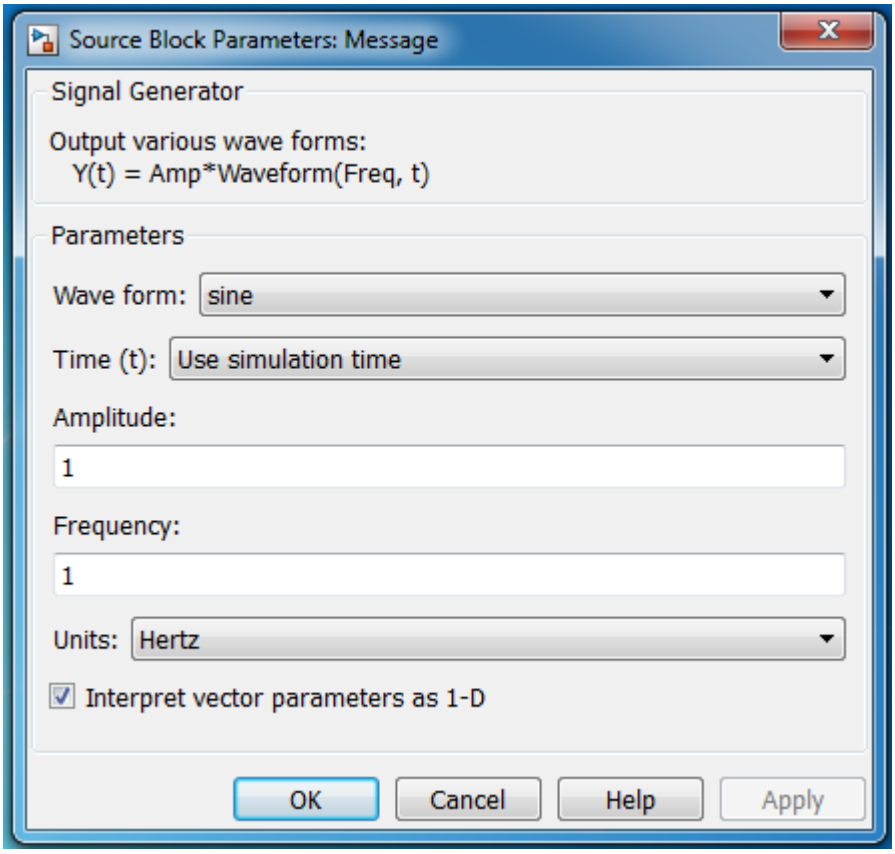
العناصر المطلوبة هي: Signal Generator, Constant, Scope, Add, Product.

سنقوم ببناء النظام على الشكل التالي:

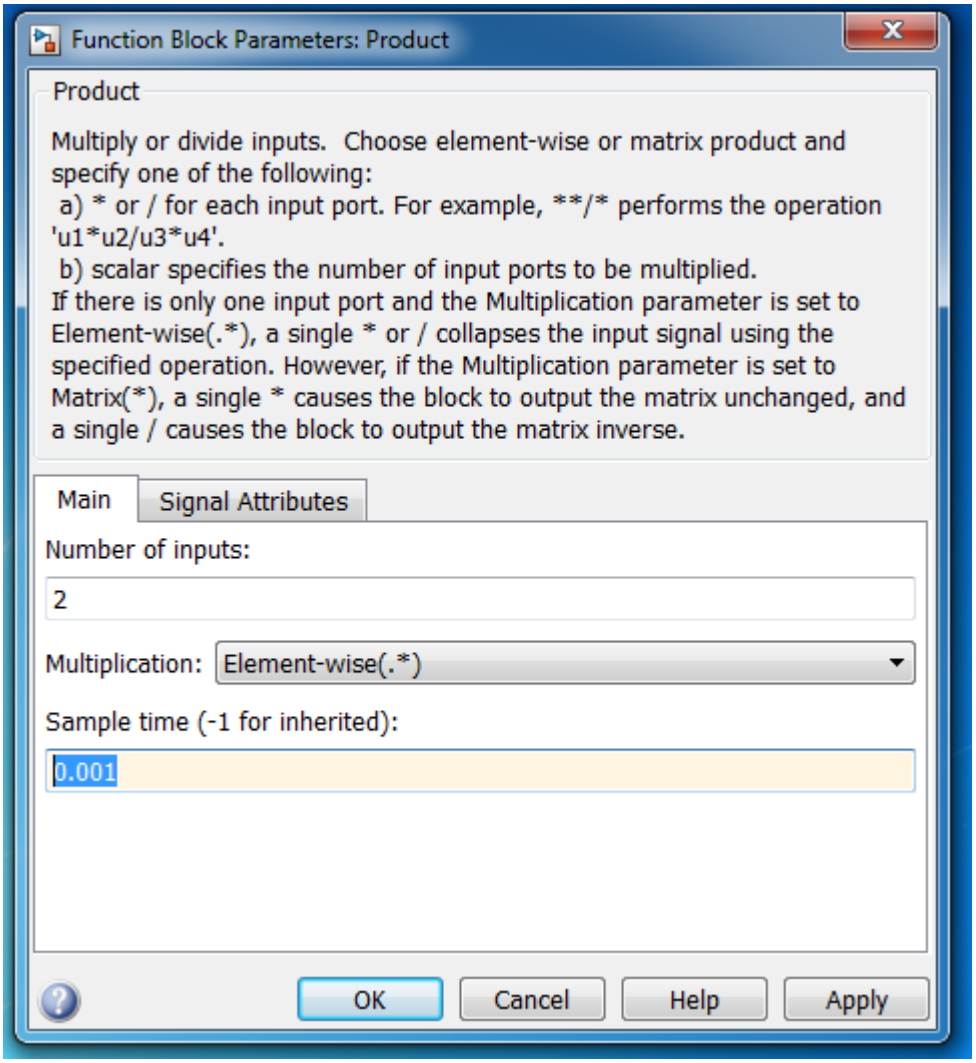


إعدادات الكتل:

- Signal Generator سيتم إعادة تسميته ب Message ومن ثم توليد إشارة Sin بمطال 1 وبتردّد 1 Hz.

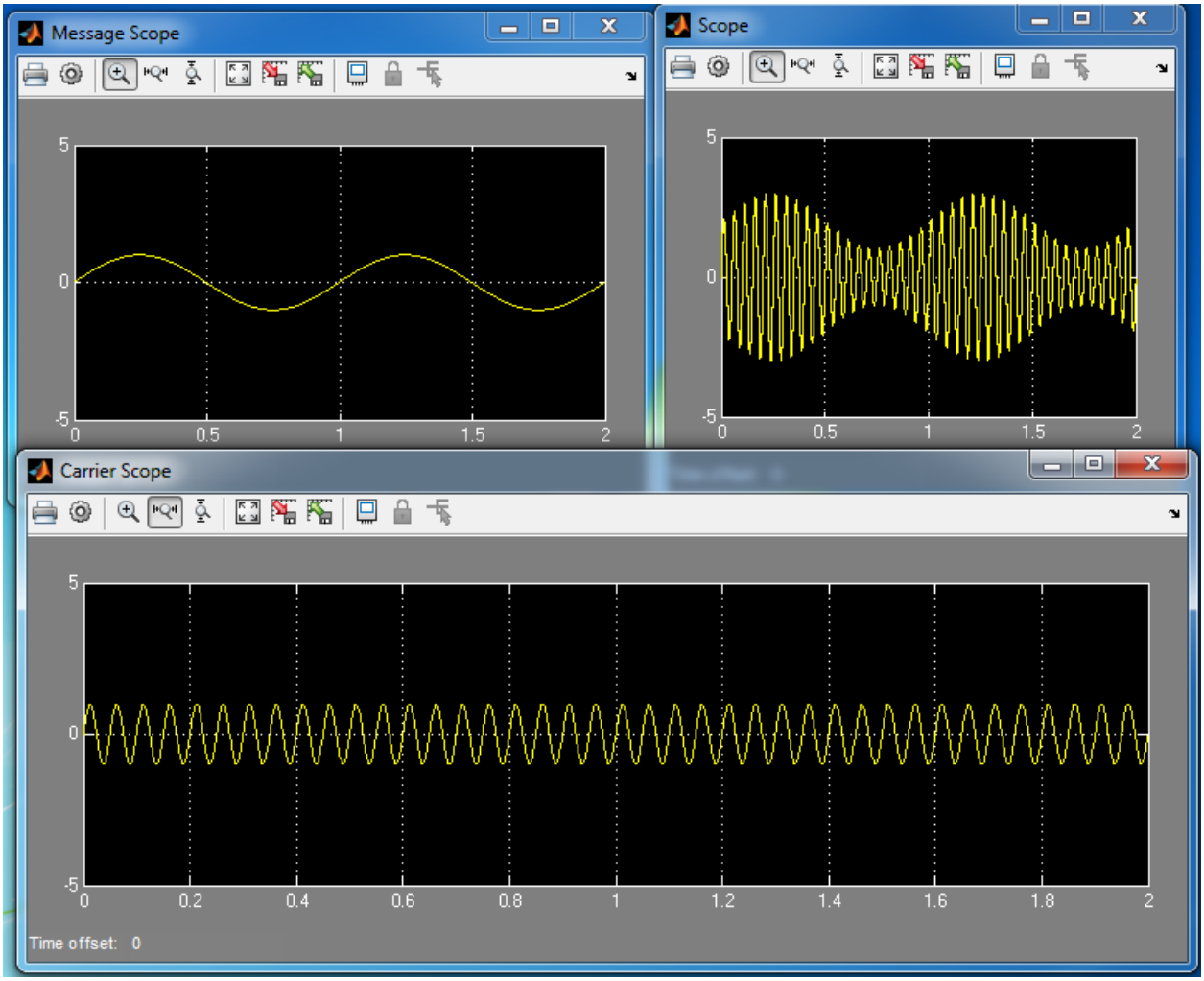


- الثابت قيمته 2.
- Signal Generator 1 سيتم تسميته ب carrier ومن ثم توليد إشارة Sin بمطال 1 وبتردّد 20 Hz (يجب ان يكون التردّد الحامل أكبر بكثير من تردّد إشارة المعلومات).
- المنظار Scope سنستخدم أكثر من منظار وكل منهم سيتم تسميته وفق مخطط النظام المعروض.
- Product سيتم تغيير Sample Time إلى القيمة 0.001 كما في الشكل:

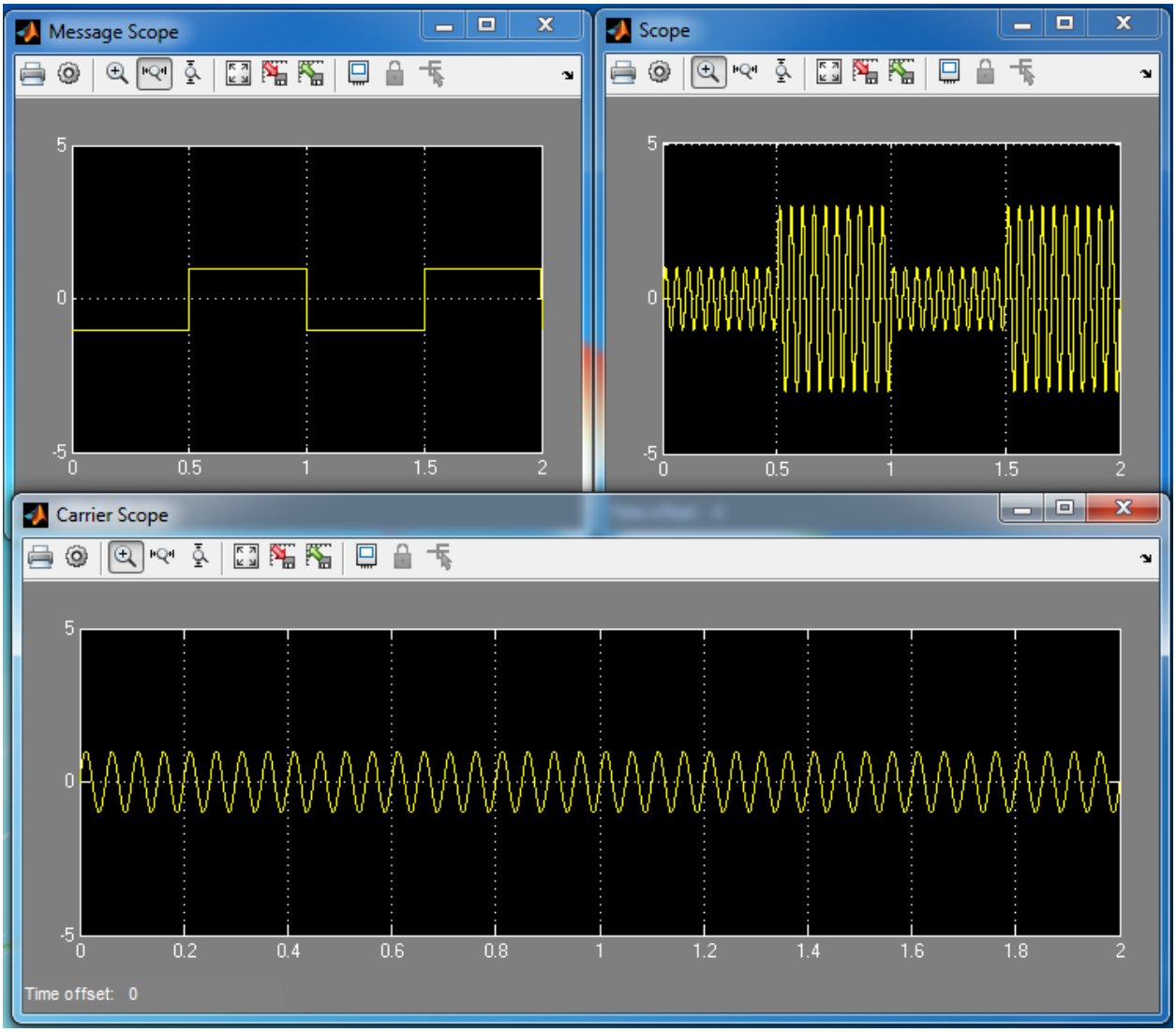


نضبط الإعدادات وفق stop time = 2 و max step size = 0.001.

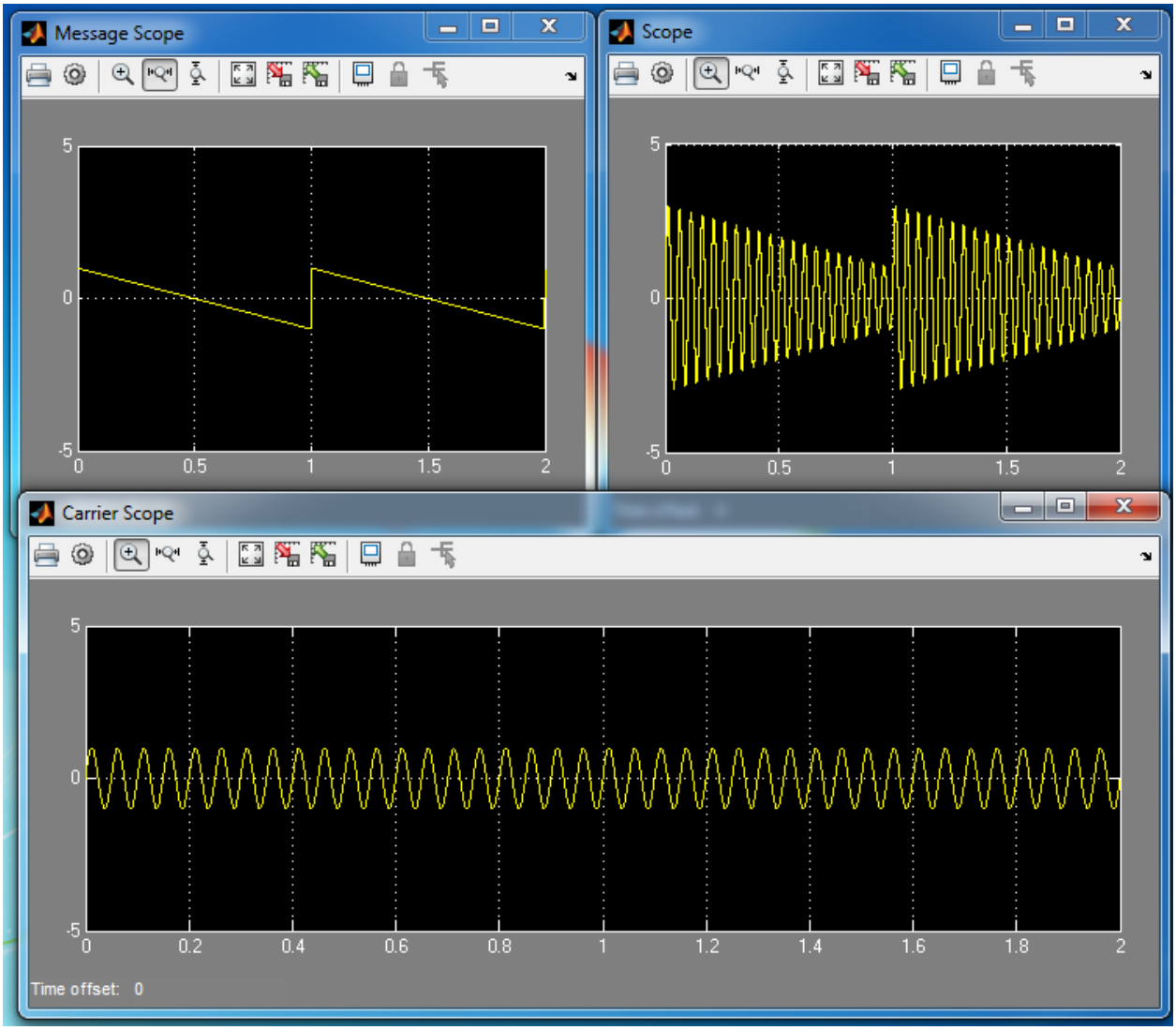
الخرج هوإشارة معدّلة مطالياً سنعرض إشارة المعلومات والإشارة الحاملة والإشارة النهائية، نلاحظ أن غلاف الإشارة المعدّلة هو إشارة المعلوامت.



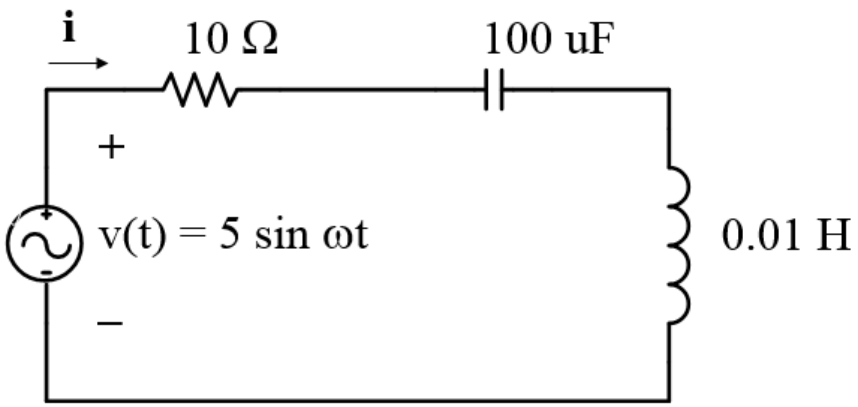
يمكن تغيير شكل إشارة المعلومات مثلاً اختر أن إشارة المعلومات هي square ستحصل على الخرج التالي.



أو إشارة سن منشار مثلاً لنحصل على الخرج التالي:



مثال: نرغب في هذا المثال في نمذجة دارة كهربائية، الدارة لها الشكل التالي:



سنقوم بإيجاد تابع التحويل للدارة:

بكتابة حلقت كيرشوف للجهد ضمن الحلقة نجد:

$$v = iR + L\frac{di}{dt} + \frac{1}{C}\int idt$$

بالاشتقاق بالنسبة للزمن نجد:

$$\frac{1}{L}\frac{dv}{dt} = \frac{di}{dt}\frac{R}{L} + \frac{d^2i}{dt^2} + \frac{i}{LC}$$

بأخذ تحويل لابلاس نجد:

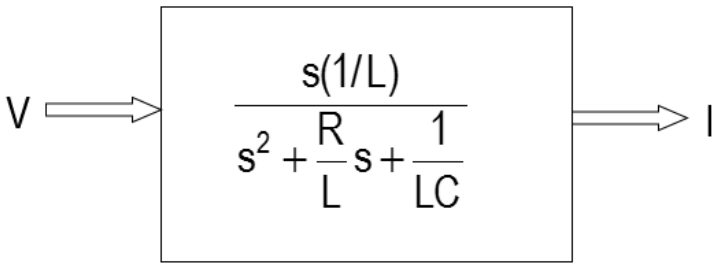
$$\frac{sV}{L} = \frac{R}{L}sI + s^2I + \frac{I}{LC}$$

$$\frac{sV}{L} = I\left[s^2 + \frac{R}{L}s + \frac{1}{LC}\right]$$

ومنه يمكن كتابة علاقة التيار على الشكل:

$$I = V\left[\frac{s(1/L)}{s^2 + \frac{R}{L}s + \frac{1}{LC}}\right]$$

وبالتالي يمكن تمثيل الدارة على الشكل التالي:



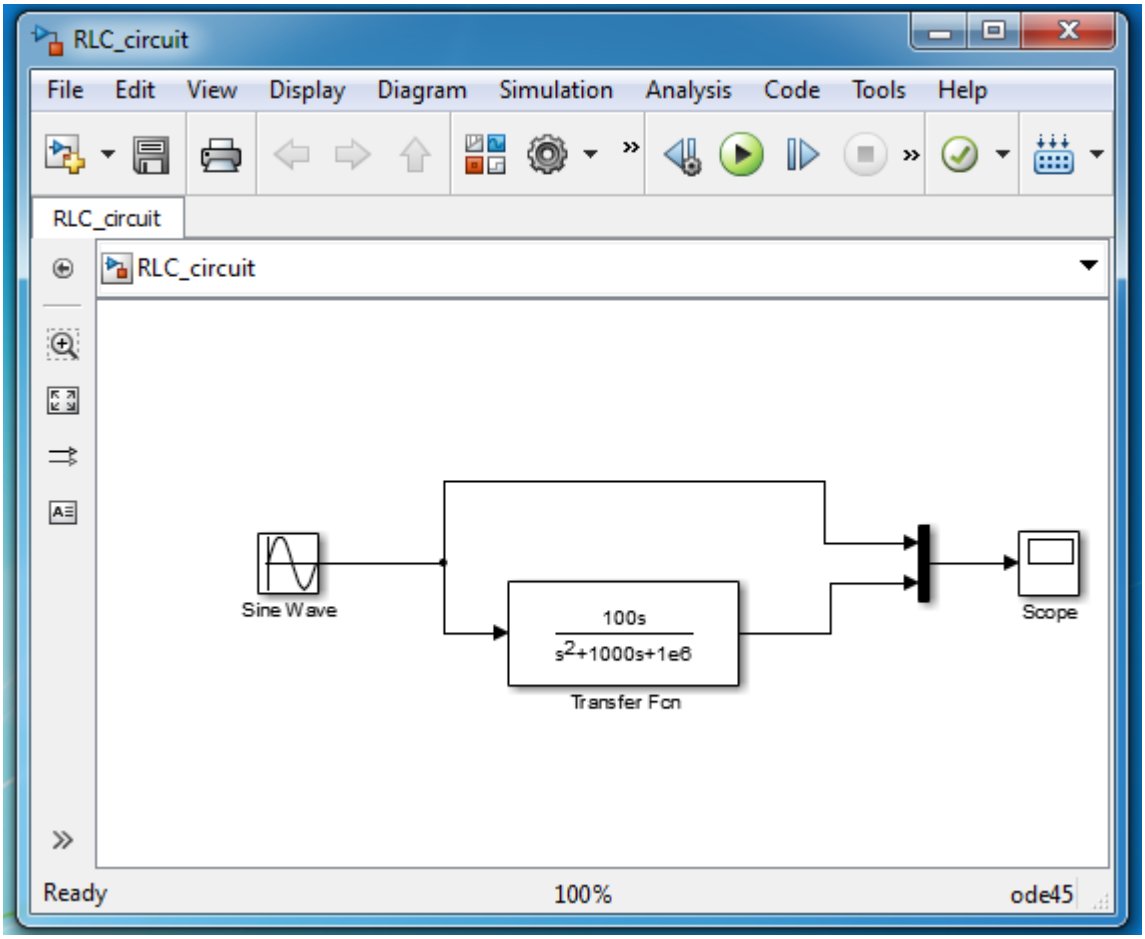
يمكن تمثيل النظام السابق بدلالة إشارة دخل إضافةً لتابع تحويل :

$$Tf = \frac{s(1/L)}{s^2 + \frac{R}{L}s + \frac{1}{LC}}$$

بالتعويض بالقيم العدّدية نجد:

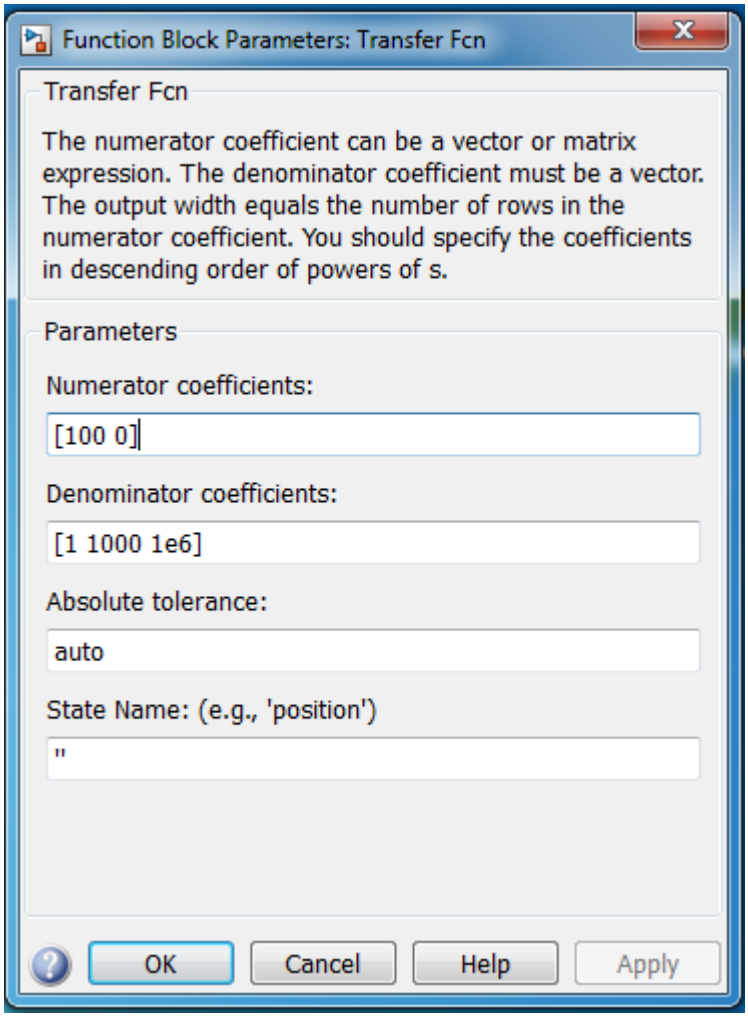
$$\frac{s(100)}{s^2 + 1000s + 1\times10^6}$$

شكل النظام ضمن Simulink:



إعدادات الكتل:

- الإشارة الجيبية بمطال 5 و بتردّد متغير سنجرب عدة قيم هي 100, 200, 500, 800, 1000, 1400, 1700, 2000.
- Transfer Function وفقاً للشكل التالي:



قم بتشغيل المحاكاة وقم بتجريب كافة القيم من أجل تردّد إشارة الدخل، ولاحظ من أجل أي قيمة يتم الحصول على المطال الأعظمي لإشارة الخرج.

مثال:

نرغب في هذا المثال، في تمثيل نظام بسيطة عبارة عن دارة مقاومة R و وشيعة L، موصولتان على التسلسل، دخل النظام عبارة عن كمون (فرق جهد) ليكن اسمه U مثلاً، وخرجه التيار i.

الهدف من هذا المثال هو التأكيد على أن Simulink أداة قوية وفعالة من أجل المحاكاة. حيث من الممكن محاكاة نفس النظام بعدة طرق.

الطريقة الأولى هو التمثيل في المجال الزمني باستخدام المعادلات التفاضلية والتي تتيح لنا تمثيل الأنظمة الخطية والغير خطية.

عن طريق تحليل الدارة باستخدام قوانين كيرشوف، نجد ان العلاقة التي تربط بين جهد الدخل وتيار الخرج هي التالي:

$$L * \frac{di}{dt} + R * i = U$$

وهي معادلة تفاضلية من الدرجة الأولى.

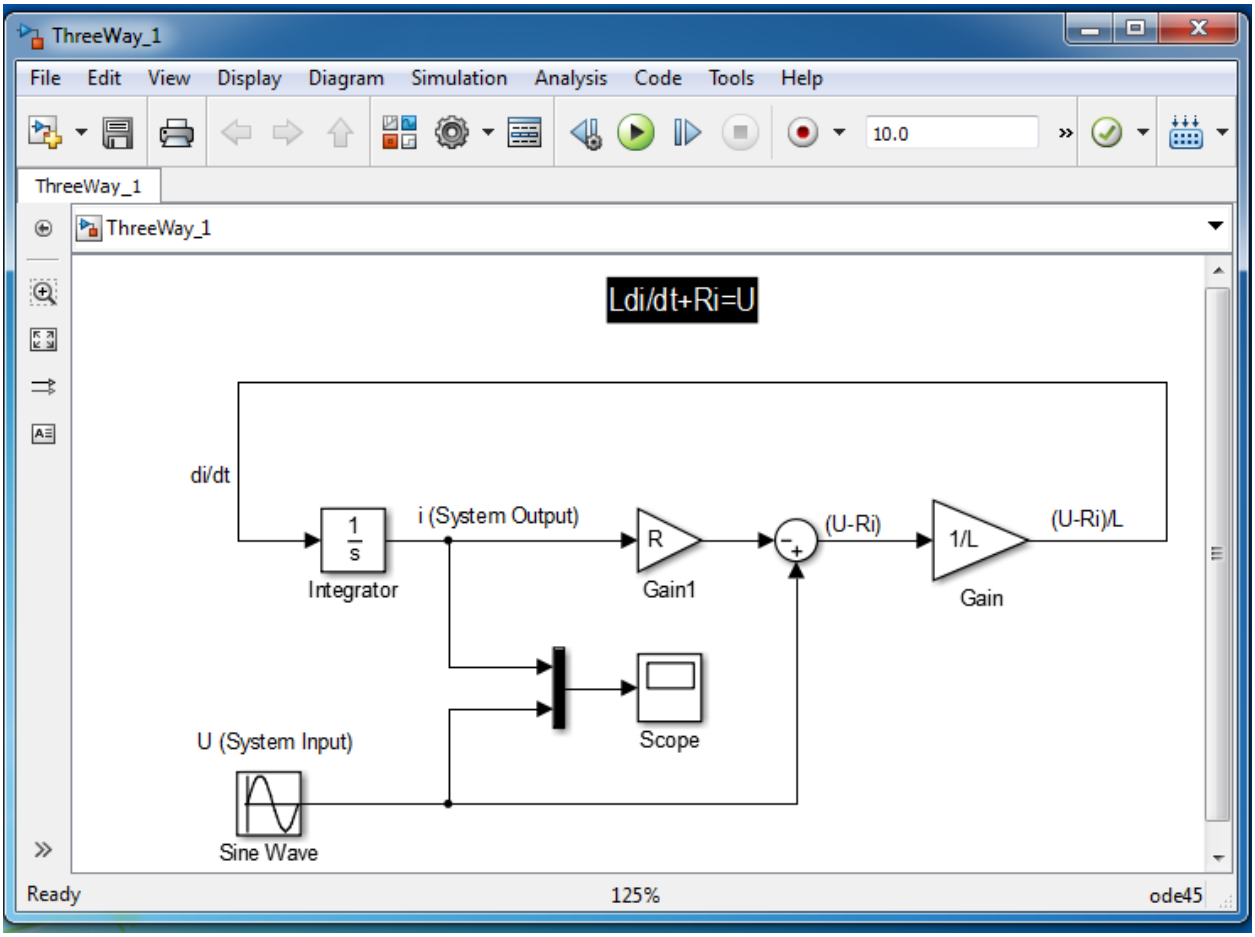
يمكن إعادة كتابة المعادلة السابقة بالشكل التالي بحيث نعزل المقدار $\frac{di}{dt}$ في طرف والطرف الآخر يحتوي على بقية الحدود:

$$\frac{di}{dt} = \frac{U - R * i}{L}$$

أولاً سنقوم بإنشاء ملف script ونكتب ضمنه الرماز التالي:

```
%% Simulink Example

clear all;close all;clc;

U=1;% Amplitude of the Sin wave

F=1;% Frequency of the input signal

R=5;% Resistor value

L=0.01;% Inductor value
```

شكل النظام باستخدام الطريقة الأولى:



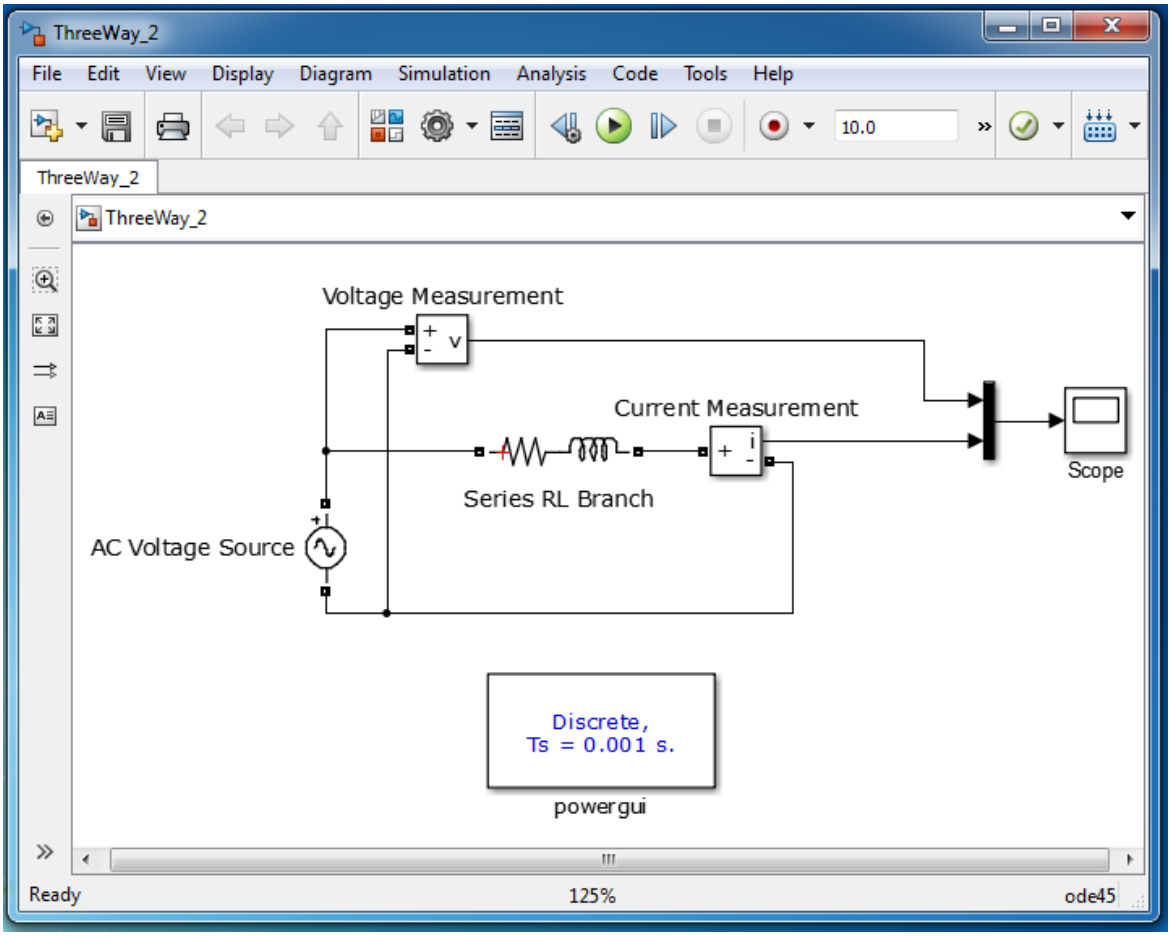
الإعدادات:

- إشارة الدخل مطال U وتردّد F
- Integrator يجب وضع القيمة البدائية 0.

- Gain الأول قيمته R والثاني قيمته 1/L.

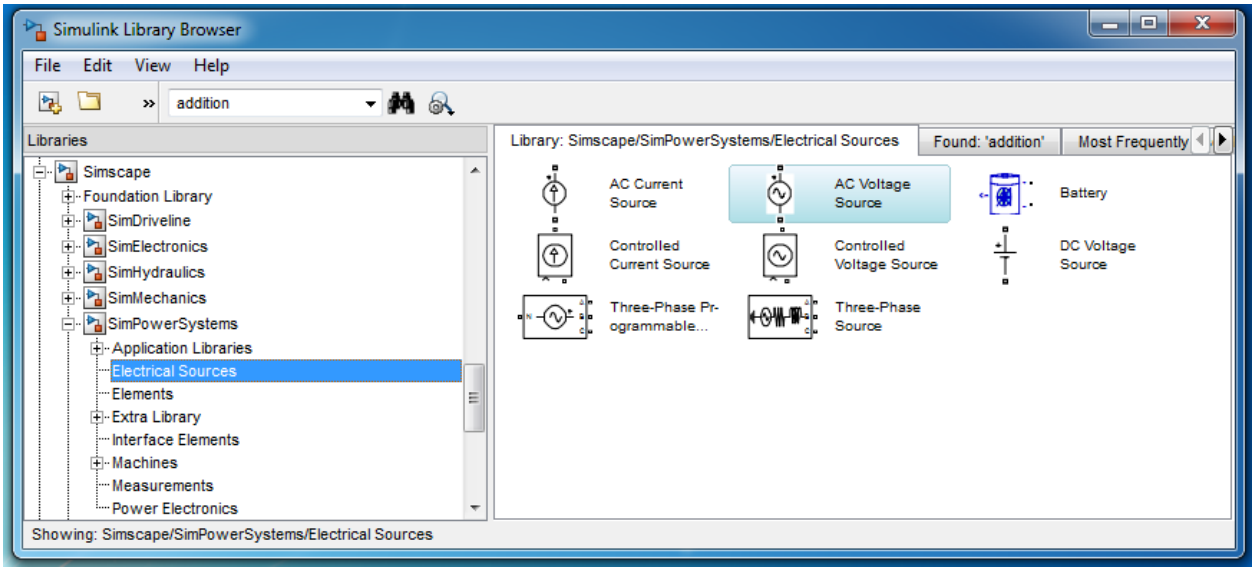
الطريقة الثانية هي التمثيل كدارة كهربائية أي وضع مقاومة ووشيعة (تمثيل فيزيائي للعناصر) وهذا الأمر متاح من SimPower Systems.

الشكل النهائي للنظام:

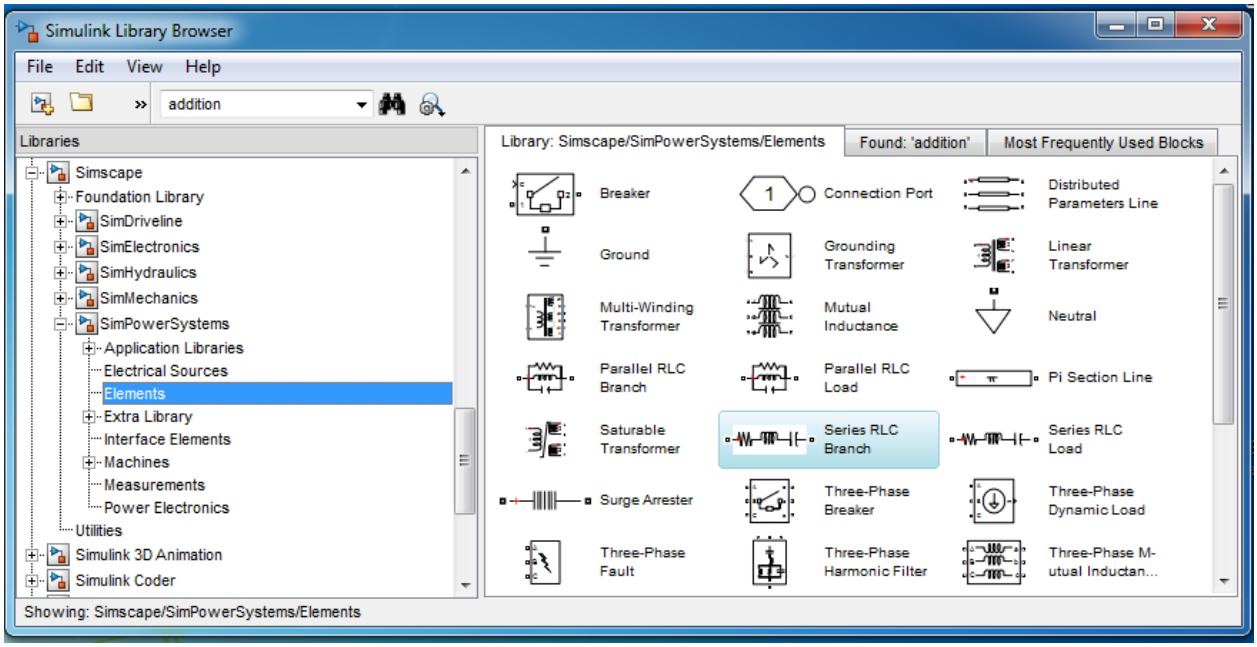


لإحضار العناصر:

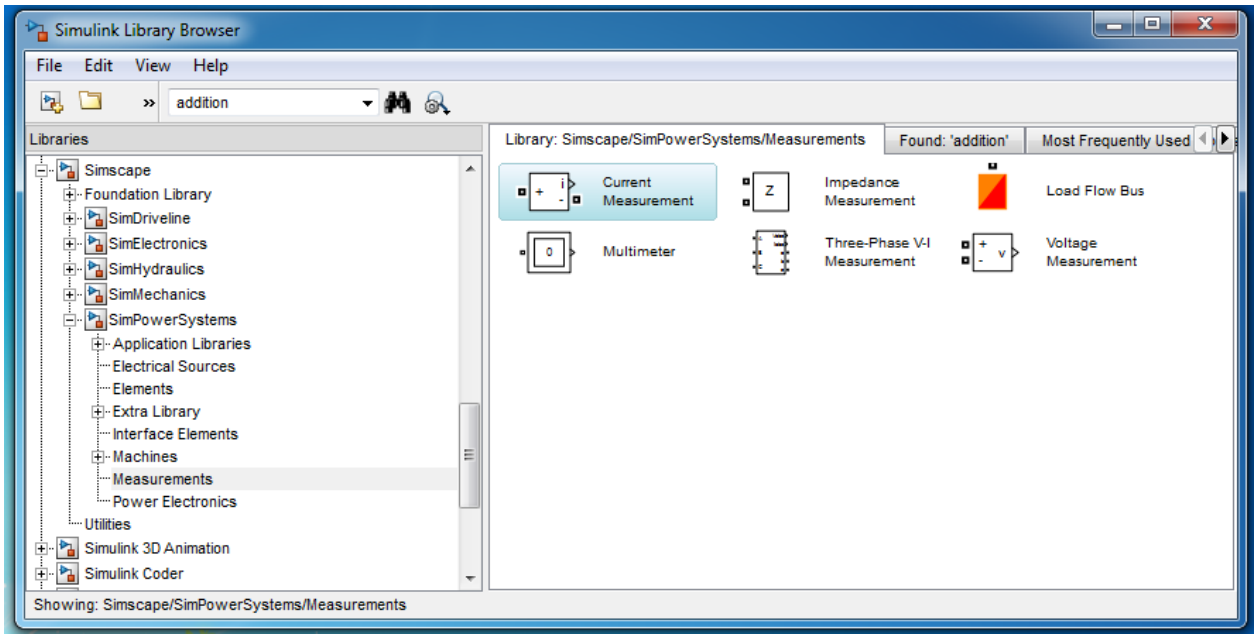
- AC Voltage Source وضع ضمنه Peak Amplitude = U و Frequency = F/(2*pi)



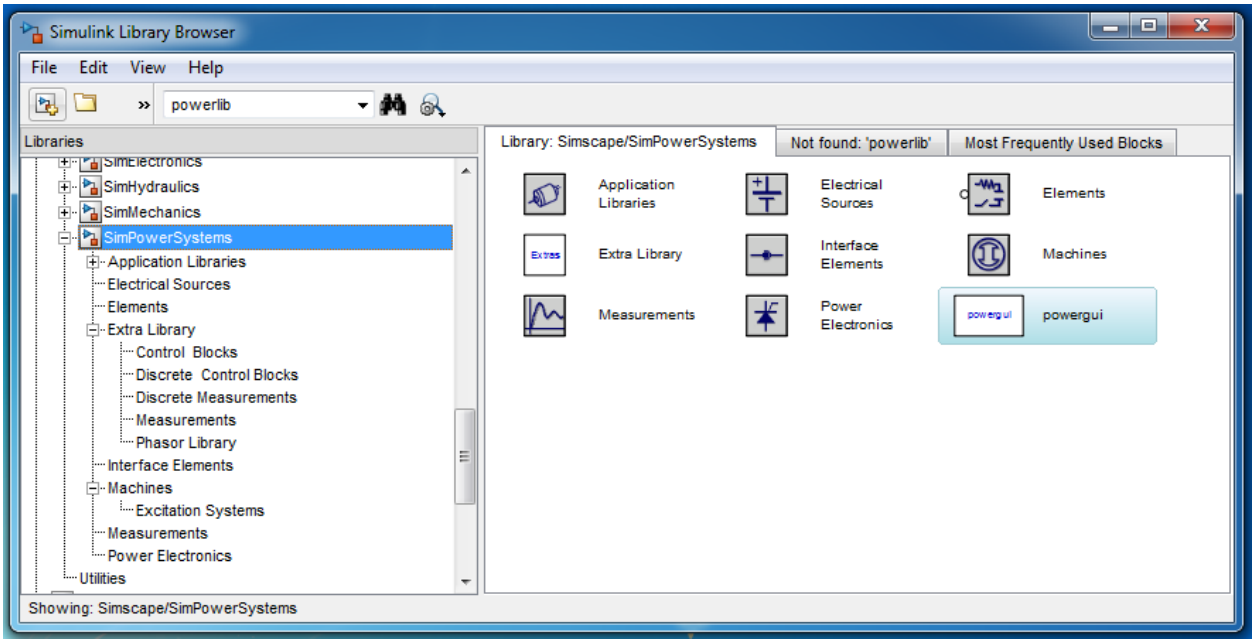
- Series RLC Branch ضمنها يمكن تحديد العناصر المرغوبة من Branch Type اختر RL فقط وضع قيمة Resistance = R و Inductance = L



- Volatge Measurement و Current Measurement.



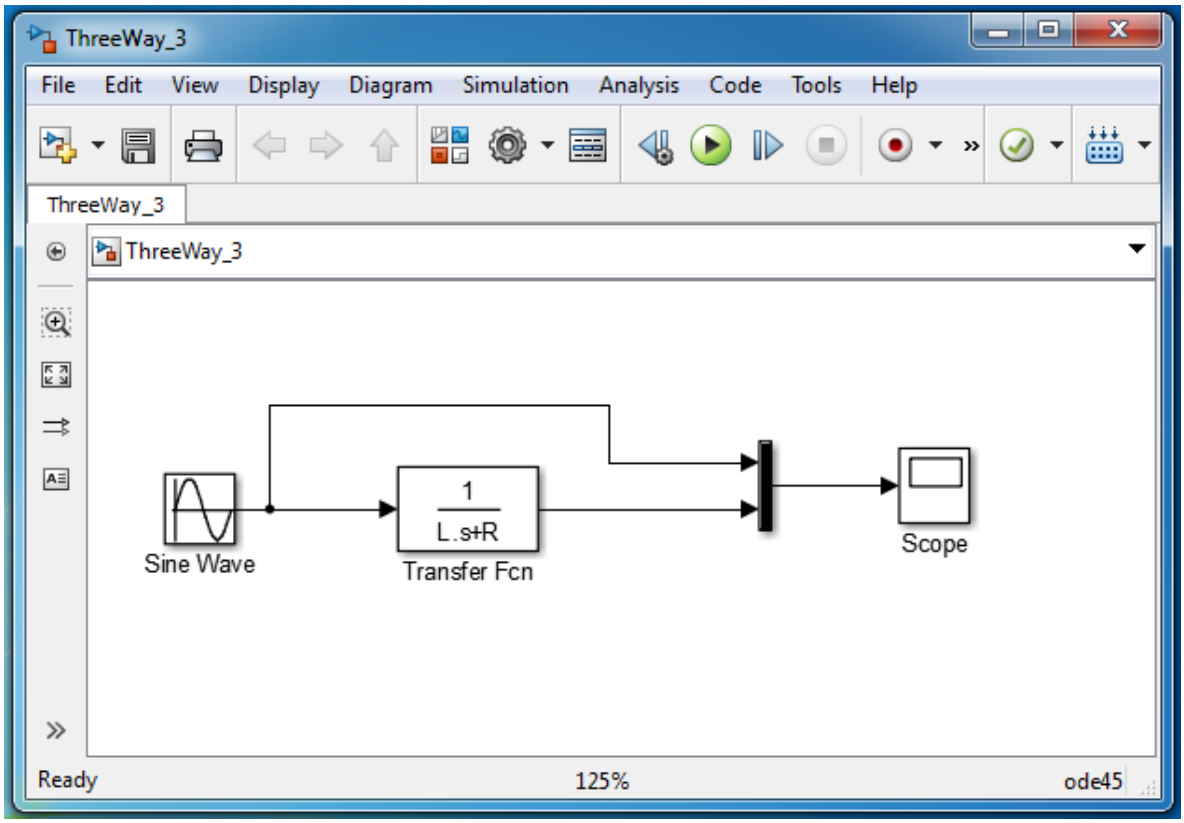
- power gui عند الضغط عليها تفتح واجهة اختر منها Configure parameters وضع داخلها Simulation Type = Discrete و Solver Type = Tusin و Sample time = 0.001



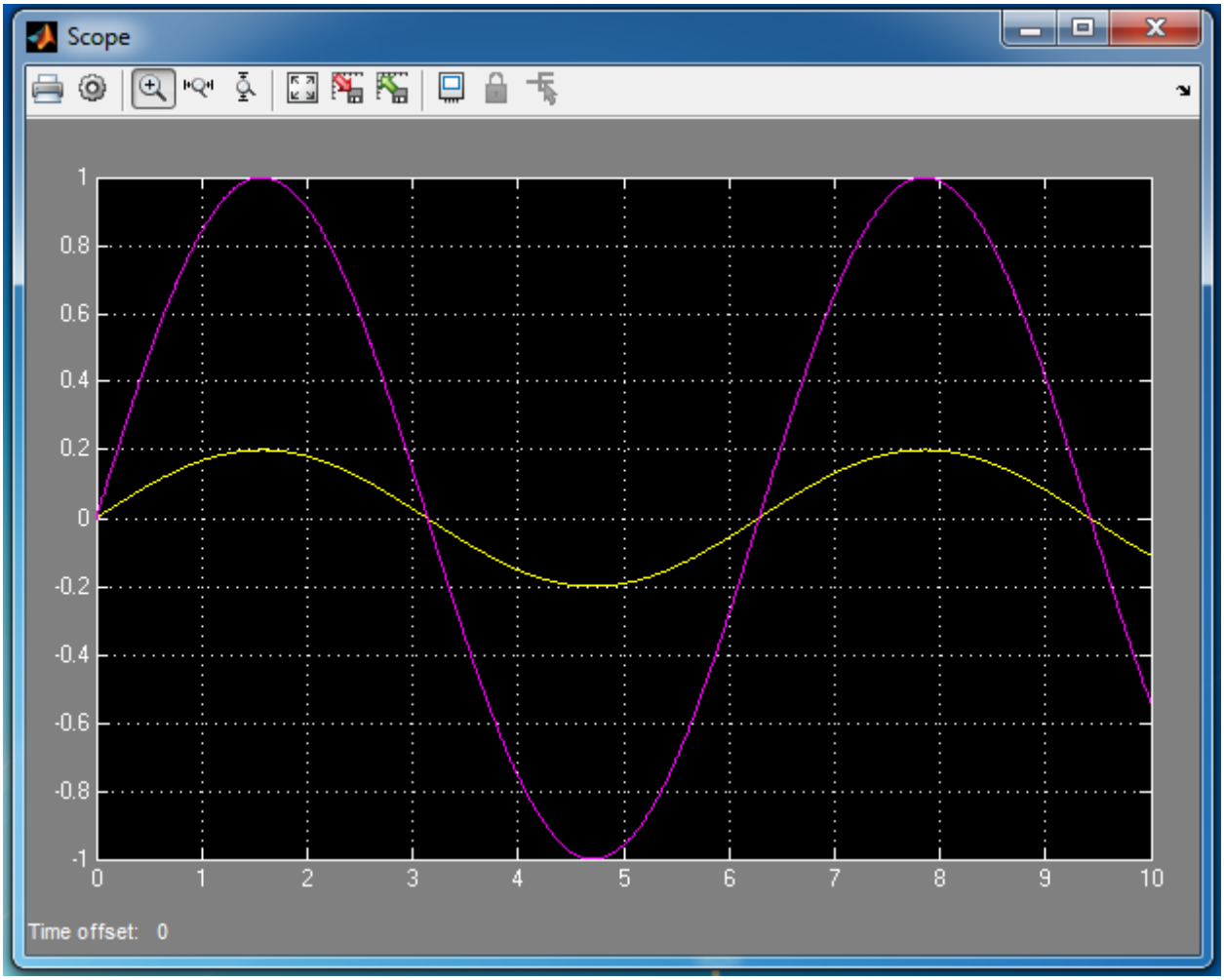
الطريقة الثالثة باستخدام تابع التحويل أي ضمن مستوي لابلاس، بتحليل بسيط للدارة نجد أم تابع التحويل هو:

$$Tf = \frac{1}{L.s + R}$$

شكل النظام هو:



عند إجراء المحاكاة للأنظمة الثلاث السابقة سنحصل على الخرج ذاته وهو على الشكل:



نلاحظ أن الخرج هو عبارة عن إشارة جيبية مختلفة في الصفحة والمطال عن إشارة الدخل نتيجةً لوجود الوشيعة والمقاومة، كما أن المطال الأعظمي للتيار هو المطال الأعظمي لجهد الدخل مقسوماً على جذر مربع قيمة المقاومة ومربع قيمة الوشيعة مضروباً بمربع النبض w.

$$i = \frac{|U|}{\sqrt{R^2 + L^2\omega^2}}$$

نلاحظ كلما ازدادت قيمة R أوتردّد إشارة الدخل أوقيمة L انخفاض في مطال إشارة الخرج. (قم بتجريب ذلك).

الفصل الثامن: الواجهات البيانية التخاطبية GUI
Graphical User Interface GUI

عنوان الموضوع:

الواجهات البيانية التخاطبية GUI

Graphical User Interface GUI

الكلمات المفتاحية:

MATLAB، واجهة بيانية تخاطبية GUI، مرشد GUIDE، Push Button، Slider، Panel، Button Group، Check Box، Radio Button، Toogle Button، Edit Text، Static Text، Pop-up Menu، List Box، Table، Axes، Toolbar.

ملخص:

سنتعرف في هذا الفصل على جزء تطبيقي جديد ضمن برمجية MATLAB® وهو بناء واجهات بيانية تخاطبية GUI.

تعتبر GUI وسيلة هامة جداً للتفاعل مع مستخدمي الحاسب بشكل عام، وكما نلاحظ تزداد الحاجات لبناء برامج فعالّة تخدم المستخدم وتوفر له مزايا وخصائص جديدة، لهذا فإن برمجية MATLAB تؤمن للمبرمج إمكانية إنشاء تطبيقات جديدة ضمن بيئة رسومية خصبة تسهل عمل المستخدم وتوفر له خيارات تصميمية واسعة عبر تعريف مسبق لمجموعة من العناصر والمتحكمات تلزم في غالبية التطبيقات ومن ثم يتاح للمبرمج بناء التطبيق بالشكل الذي يرغب به عن طريق تصميم الواجهات وتوصيف عملها عبر كتابة رماز برمجي وفقاً للحاجات والمتطلبات؛ كما ان برمجية MATLAB تتيح لمستخدم بناء عناصر جديدة برمجياً خاصة به.

سنعرض في هذا الفصل مقدمة تعريفية عن الواجهات البيانية التخاطبية، ونوضح آلية العمل وطريق بناء الواجهات ضمن MATLAB، ثم سننتقل لتصميم الواجهات باستخدام مرشد GUIDE متاح من MATLAB، حيث سنعرض بعض الأمثلة الهامة بعد التعرف على العناصر المعرفة ضمن GUIDE وأهميتها.

أهداف تعليمية:

يتعرف الطالب في هذا الفصل على:

- يتعرف على الوجهات البيانية التخاطبية GUI ضمن MATLAB.
- ينجز بعض التطبيقات البسيطة باستخدام GUI.
- يمتلك أداة قوية لبناء تطبيقات فعالة ومعقدة.
- يستخدم جميع المعارف السابقة المكتسبة ضمن المقرر لتنجيز برامج مفيدة بمساعدة GUI.

مقدمة عن الواجهات البيانية التخاطبية GUI Introduction on Graphical User Interface GUI

تتميز برمجية MATLAB، واعتماداً على لغة MATLAB بأنها تحتوي على الأدوات الأساسية والضرورية لتشكيل تطبيق برمجي متكامل، بشكل بسيط وبعيد عن تعقيدات الإظهارات، وبحيث لا يشغل المبرمج أو المستخدم نفسه بحل مشاكل تأمين أزرار أوامر مثلاً، وإنما تكون المهمة الأساسية هي الاستفادة من الأدوات الموجودة، وربطها مع الإجراءات اللازمة لتحقيق الهدف من البرنامج أو التطبيق.

عند العمل على تطوير برنامج Program أو تطبيق Application يوجد عدد كبير من المعايير الواجب اتخاذها مثل سهولة الاستخدام، الوضوح، صحة الخرج، .... يمكن للمبرمج التعرف عليها عن طريق تصفح مقالات مختصة في هذا الموضوع، الهدف في هذا الفصل هو تعلم بناء الواجهات التخاطبية طبعاً مع احترام القواعد السابقة.

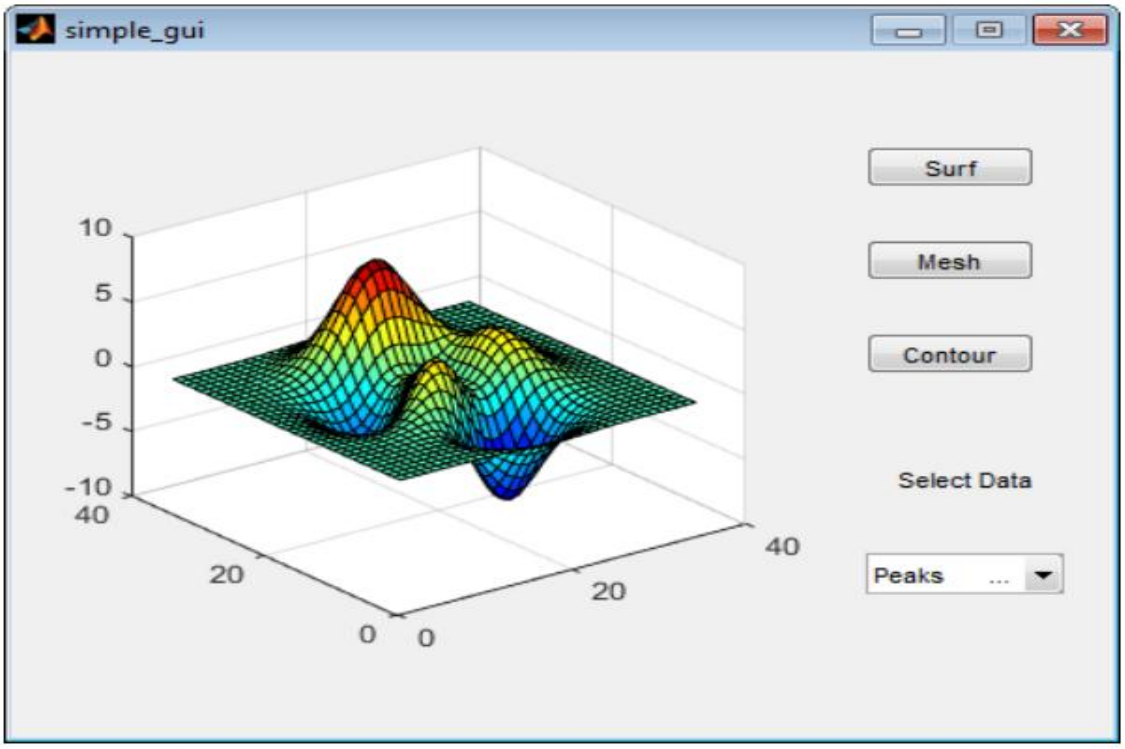
ما هي الواجهات التخاطبية ؟؟ What is a User Interface UI??

تعتبر الواجهة التخاطبية وسيلة بيانية متضمنة في نافذة أو أكثر، تحتوي كل منها على متحكمات تدعى عناصر components، تتيح للمستخدم القيام بمهام تفاعلية.

لا يحتاج المستخدم عند استخدام الواجهات البيانية التخاطبية GUI ، من أجل القيام بالمهمة المطلوبة أو من أجل الحصول على خرج ما، لإنشاء ملف script أو كتابة تعليمات ضمن Command Window.

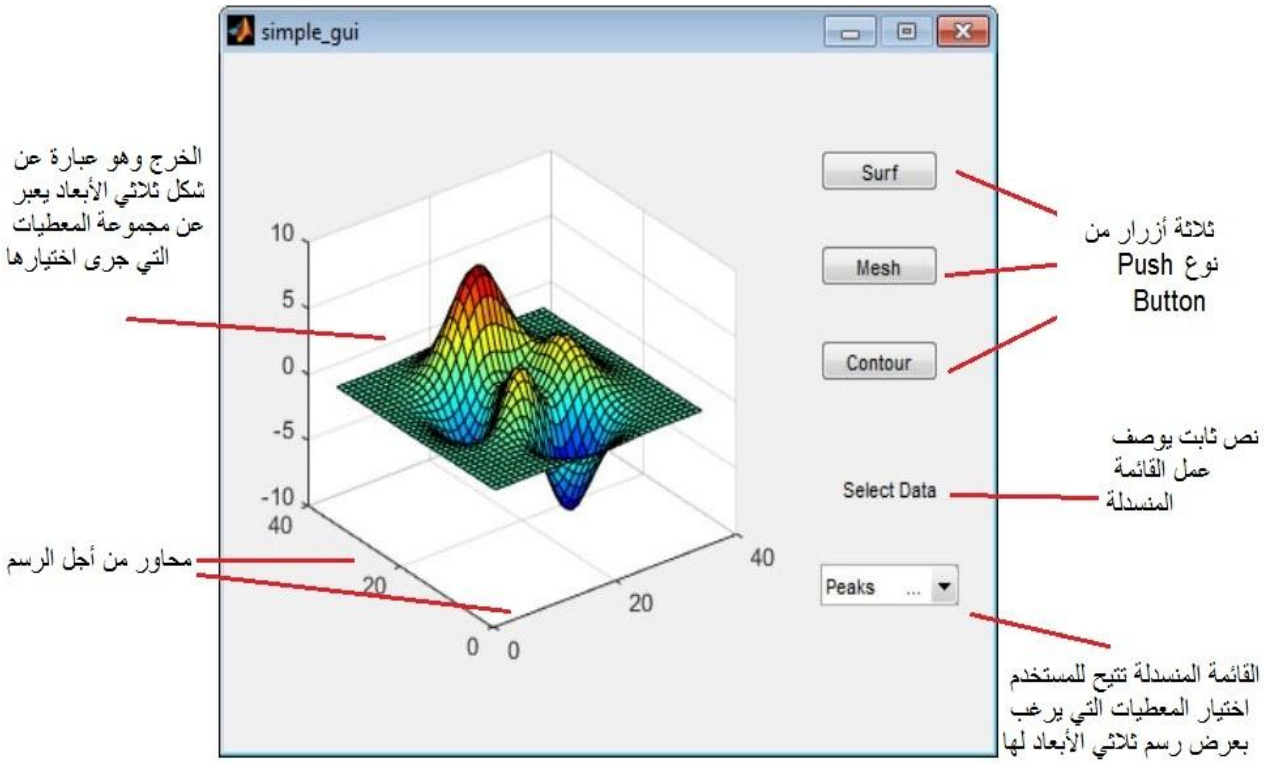
تتميز الواجهات التخاطبية ضمن MATLAB بسهولة التعامل معها، إضافةً إلى الإمكانيات الهائلة التي تتمتع بها، كما انها تتيح القيام بأي نوع من الحسابات العدّدية. وتدعم القراءة والكتابة من/ إلى ملفات معطيات، التواصل مع واجهات تخاطبية أخرى، وعرض المعطيات سواء عن طريق الرسوم والمنحنيات أو عن طريق الجداول.

يوجد عدد كبير من العناصر الأساسية ضمن أي واجهة تخاطبية UI، والتي طبعاً تختلف حسب حاجة المستخدم أو التطبيق، يبين الشكل التالي واجهة بيانية بسيطة يمكن بناؤها بسهولة باستخدام MATLAB،



تتألف الواجهة السابقة من العناصر التالية:

- عنصر محاور Axes Component.
- قائمة منسدلة pop-up menu تعرض ثلاث مجموعات من المعطيات تتوافق مع توابع MATLAB وهي:
  - Peaks.
  - Membrane.
  - Sinc.
- عنصر نص ثابت Static Text Component من أجل توصيف القائمة المنسدلة أي توضيح عملها.
- ثلاثة أزرار Buttons تؤمن للمستخدم أنواع مختلفة من الرسومات plot وهي:
  - Surface.
  - Mesh.
  - Contour.



يتم اختيار نوع من أنواع المعطيات عن طريق القائمة المنسدلة، وعند الضغط على زر ما من Push Button يتم عرض هذا النوع من المعطيات عن طريق أحد الرسومات ثلاثية الأبعاد 3-D Plot المختار عبر الزر.

How does a User Interface UI work ?? كيف تعمل الواجهات التخاطبية ؟؟

عادةً، تنتظر الواجهات البيانية أي حدث event يقوم به من المستخدم، وبعدها تستجيب لكل من الأحداث السابقة على حدى. الأحداث هي ما يقوم به المستخدم من أفعال عند استخدام الواجهة البيانية كالنقر بزر الفأرة على زر أوامر ما أو النقر المزدوج لأيقونة ما أو اختيار أمر من قائمة أو الضغط على لوحة المفاتيح.

بعد وقوع حدث ما يقوم نظام التشغيل بتنفيذ إجراء مرتبط مع الحدث الواقع ومع الأداة التي وقع عليها الحدث ضمن نافذة الواجهة التخاطبية، لذلك توصف برمجة الواجهات بأنها برمجة مسيرة بأحداث event-driven.

إن المناقشة السابقة تدفعنا إلى النتيجة التالية: يترتب على المبرمج تجزئة البرنامج (التطبيق، الواجهة) إلى إجراءات يستجيب كل إجراء منها لحدث معين.

بشكل عام، تبنى الواجهات التخاطبية عن طريق ربط كل عنصر (متحكم) مع تابع يدعى callbacks، أو عدة توابع، سميت هذه التوبع ب callbacks من حقيقة أنه يتم الرجوع إليها أو استدعائها "call back" لمعرفة ما هو الفعل الواجب تنفيذه عند إنشاء حدث event ما ناتج عن الضغط على عنصر ما.

Callback هو تابع يتم كتابته وربطه مع عنصر محدّد ضمن الواجهة البيانية التخاطبية GUI، يقوم تابع Callback بالتحكم بالواجهة أوسلوك العنصر عن طريق تطبيق فعل ما بعد وقوع حدث مرتبط بالعنصر، مثلاً

عن الضغط على زر يقوم التابع بتنفيذ عملية الجمع بحيث يأخذ قيم العدّدين المراد جمعها من خلال حقلين ضمن الواجهة ومن ثم ينفذ عملية الجمع ويعرض الناتج في حقل جديد.

طرق بناء واجهات بيانية ضمن MATLAB Ways to build MATLAB User Interfaces UIs

الواجهة البيانية ضمن MATLAB هي عبارة عن نافذة شكل figure window يتم إضافة لها عناصر وفقاً لحاجة أو رغبة المستخدم، المبرمج أو التطبيق. يتم تحديد حجم، موضع، وشكل العنصر المضاف وفقاً للخيارات المستخدم التصميمية ومن ثم يتم توصيف عمل كل عنصر باستخدام توابع Callbacks.

يمكن بناء الواجهات البيانية ضمن MATLAB بطريقتين هما:

1. إنشاء الواجهة البيانية برمجياً.

باستخدام هذه الطريقة يتم بناء الواجهات بالكامل برمجياً، بحيث يعرف الرماز المكتوب خصائص وسلوك مكونات الواجهة وهو عبارة عن ملف ذو لاحقة m. ، ويمكن تشغيل الواجهة عن طريق تنفيذ الرماز Run.

لن نقوم بتطوير الواجهات باستخدام هذه الطريقة نظراً لصعوبتها مقارنةً بالطريقة الثانية نسبةً لطلاب لم يستخدموا برمجية MATLAB مسبقاً أو بشكل كبير ولم تتشكل لهم الخبرة الواسعة في مجال الواجهات التخاطبية..

2. إنشاء الواجهة البيانية باستخدام المرشد GUIDE.

يتم ضمن هذه الطريقة البدء مع شكل figure فارغ يقوم المستخدم بإضافة العناصر إليه وذلك بالاستعانة بمحرّر تصاميم بيانية هو المرشد GUIDE، يقوم GUIDE بإنشاء وربط رماز برمجي يحتوي على callbacks للواجهة البيانية وعناصرها، يحفظ GUIDE الواجهة على شكل ملفين الأول هو الشكل يعبر عن الواجهة الرسومية و يحتوي العناصر المكونة لها (بملف ذو لاحقة fig.) والثاني هو الرماز البرمجي أو الأوامر البرمجية التي تؤدي إلى تنفيذ البرنامج التطبيقي المطلوب تحقيقه (بملف ذو لاحقة m.). ويمكن تشغيل الواجهة من كلا الملفين.

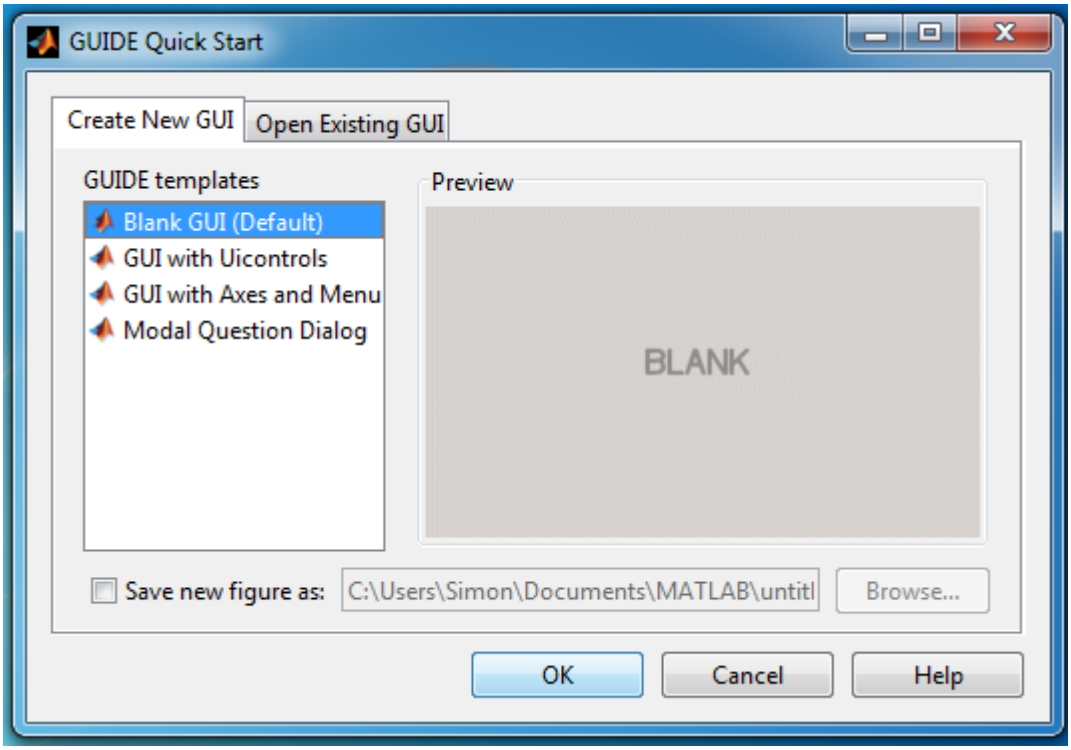
يجب التنويه إلى إمكانية إنشاء UI باستخدام GUIDE ومن ثم تعديلها برمجياً، إلا أنه لا يمكن إنشاء UI برمجياً ومن ثم التعديل عليها باستخدام GUIDE.

بدء العمل مع الواجهات البيانية التخاطبية GUI Getting Start with GUI

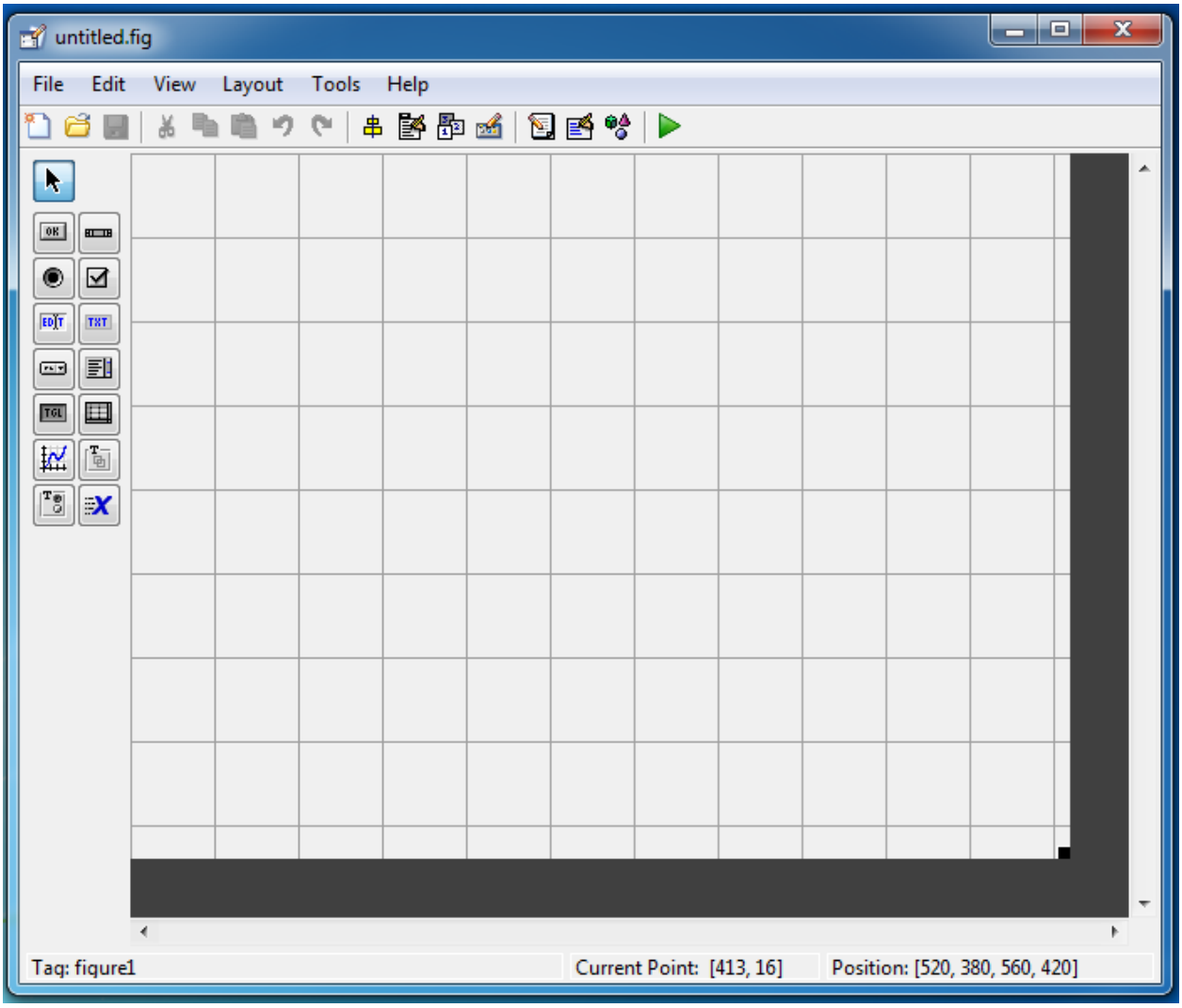
فتح واجهة بيانية جديدة باستخدام محرّر GUIDE Open a New UI in the GUIDE Editor

يمكن الوثول إلى GUIDE ضمن MATLAB عن طريق كتابة التعليمة guide ضمن Command Window، أو يمكن استخدام Toolstrip

اضغط على New ستظهر أمامك لائحة، اختر منها Graphical User Interface، سيظهر أمامك الواجهة التالية.



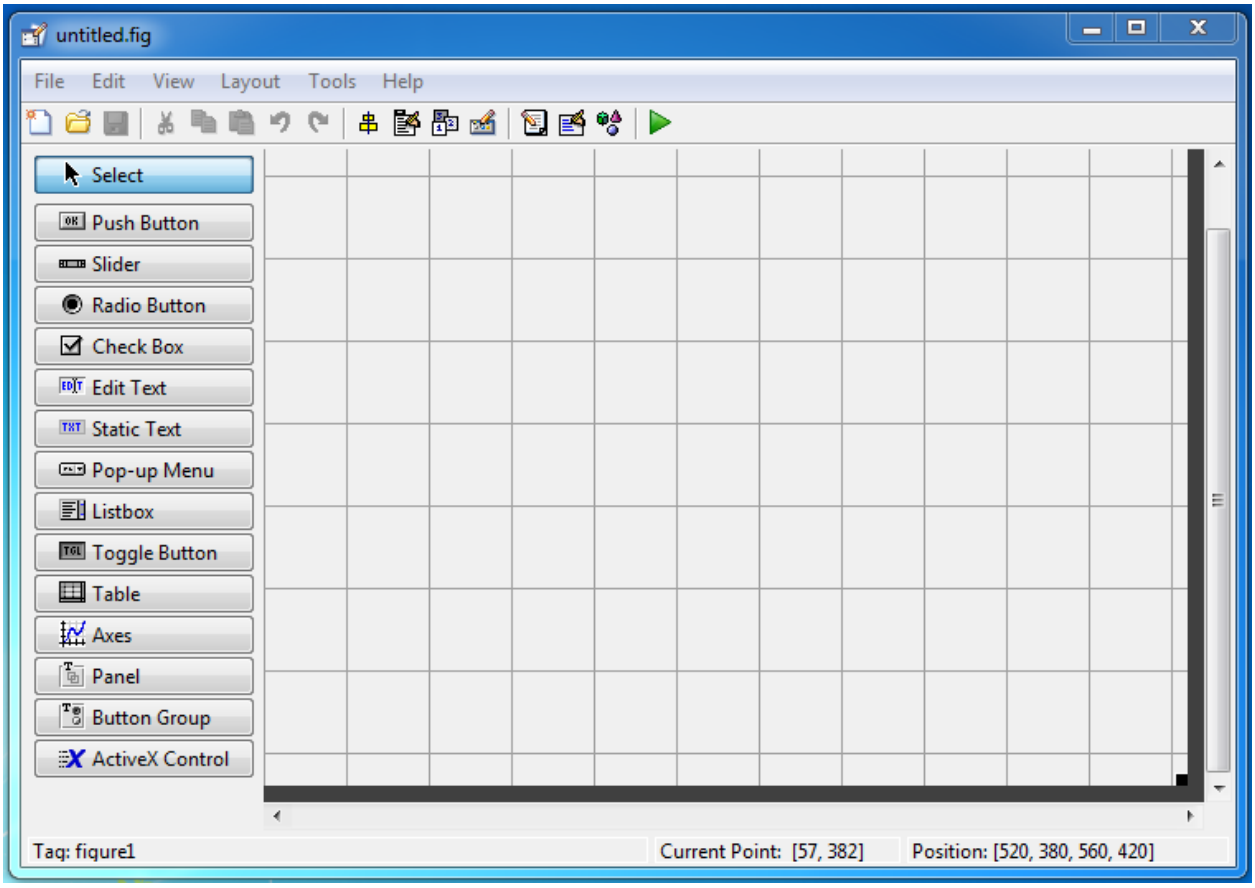
عادةً يتم اختيار Blank GUI من أجل البدء بواجهة فارغة، بعد الضغط على OK نحصل على الواجهة التالية:



تحتوي الواجهة السابقة على جميع العناصر components الممكن إضافتها إلى الواجهة، من أجل عرض أسماء مكونات الواجهة البيانية ضمن الواجهة السابقة، نقوم بما يلي:

1. اختر File → Preferences → GUIDE
2. اختر Show names in component palette
3. اضغط OK

تحصل عندها على الشكل التالي:



التعرف على العناصر المتاحة لبناء واجهة بيانية

Getting to know available components to design a UI

يبين الشكل التالي جميع العناصر المتاحة ضمن GUIDE من أجل بناء واجهة بيانية UI ضمن MATLAB، سنعرض ضمن جدول هذه العناصر إضافةً إلى الأيقونة الخاصة بكل عنصر ووصف له.

العنصر	الأيقونة	الوصف
Push Button		عند الضغط على مثل هذا الزر يتم توليد حدث ما، مثلاً لنعتبر أنه زر OK عند الضغط عليه يتم تطبيق إعدادات معينة بعد اختيارها أو يتم إغلاف الواجهة مثلاً
Slider		هو عبارة عن زر قابل للانزلاق بين قيمتين عدديتين بخطوة معينة، يتم تحريك المزلاق Silder عند طريق الضغط عليه وسحبه، مثلاً يستفاد منه في حالة الحاجة لتغيير قيمة عنصر ضمن الواجهة لمعرفة أثره مثلاً على منحني تابع.
Panel		هي عبارة عن لوحة، تفيد في تنظيم العناصر ضمن الواجهة ضمن مجموعات، تفيد panel في جعل الواجهة أبسط للفهم وأكثر وضوحاً ويوجد لكل لوحة عنوان وحدود، مثلاً عند تنفيذ واجهة تقوم عمل الآلة الحاسبة يمكن تجميع عمليات الجمع والضرب والقسمة والطرح ضمن مجموعة وتسميتها العمليات الحسابية، ومن الممكن إنشاء لوحة جديدة تحتوي على الأزرار وتسميتها أزرا إضافةً للوحة تحتوي على حقول لإدخال الأرقام مثلاً كخيار إضافي إن عملية الفصل السابقة تجعل الواجهة أفضل للمستخدم من ناحية سهولة الاستخدام.
Button Group		تشابه Panels إلا أنها تستخدم لتنظيم عملية اختيار ضمن عدة خيارات متاحة من عدد من الأزرار مثل radio buttons و toogle buttons.
Check Box		يمكن لهذا النوع من الأزرار توليد حدث عندما يتم اختياره، كما يمكن معرفة حالته أي إزا تم اختياره أو لم يتم اختياره، يستفاد منه عندما يتم تزويد المستخدم بعدد من الخيارات المستقلة ويطلب من المستخدم اختيار منها، مثلاً اختيار المواد التي يرغب في دراستها ضمن الفصل الحالي.
Radio Button		مشابهة ل check box إلا أنها تستخدم عادةً ضمن عدد معين من الأزرار من نفس النوع Radio Buttons والمرتبطة مع بعضها، مثلاً نرغب بجعل المستخدم اختيار مجال لسعر المنتج الذي يود بعرضه، عند الضغط على أحد هذه الأزرار يتم إلغاء اختيار أي زر آخر ضمن نفس المجموعة، ينصح باستخدام Button Group عند استخدام مجموعة من Radio Buttons.

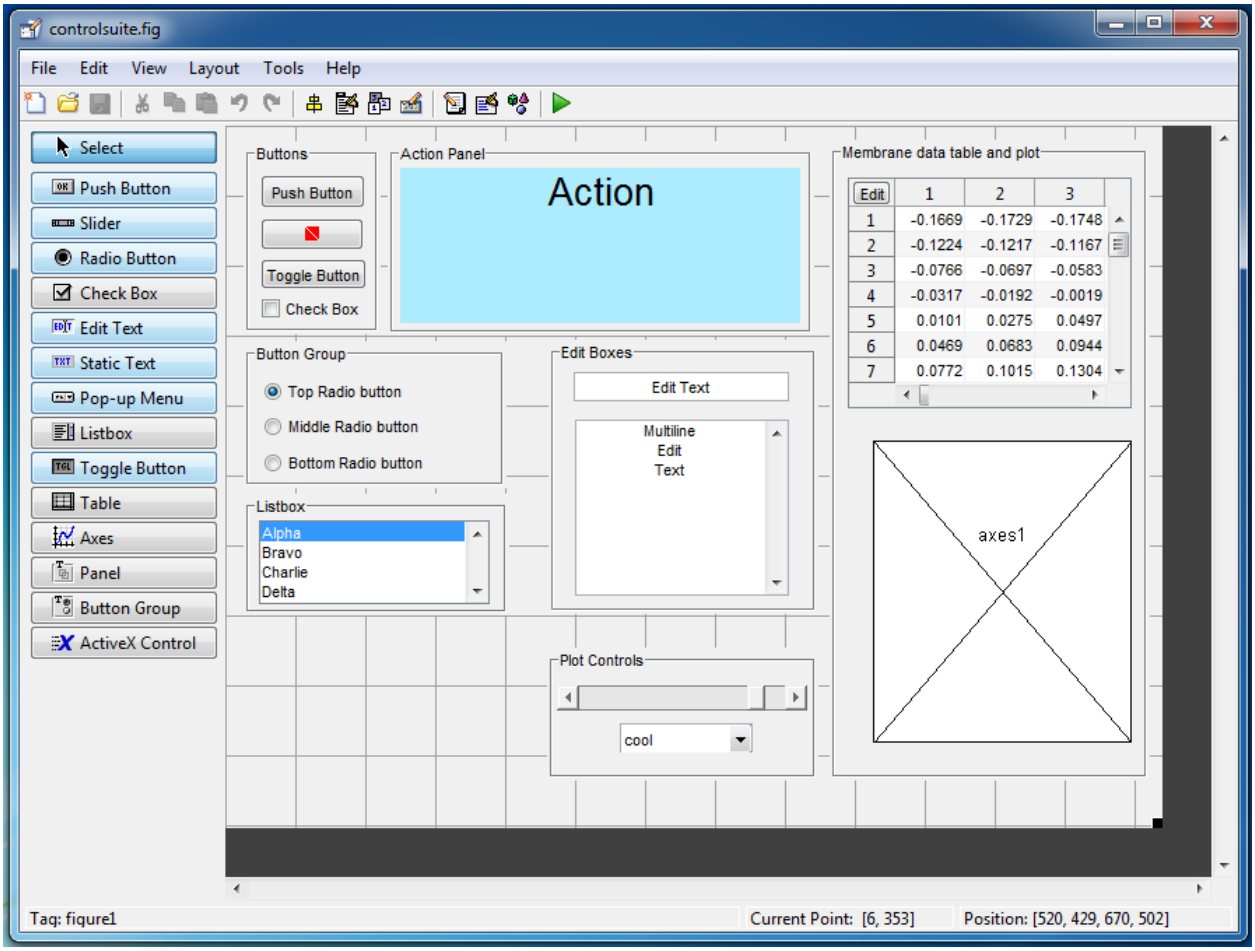
Toogle Button		هذا النوع من الأزرار يقوم بتوليد حدث ويؤشر إلى حالته فيما إزا تم ضغطه أم لا أي إزا كان On أو Off، يحافظ الزر على شكله أنه مضغوط أي حالته On إلى حين الضغط عليه مرة أخرى لينتقل عندها إلى الحالة Off، ينصح باستخدام Button Group عند استخدام مجموعة من Toogle Buttons.
Edit Text		هو عبارة عن حقل يتيح للمستخدم بكتابة نص ضمنه أو تعديل نص ما، يمكن استخدامه كدخل للواجهة مثلاً يريد المستخدم إدخال رقم لحساب ناتج العاملي factorial له. تجدر الملاحظة أنه يجب تحةيل الأعداد المدخلة داخل الرماز البرمجي إلى أعداد لأنه يفهم الدخل على أنه محرف char او سلسلة محرفية stringيمكن استخدام توابع str2num.
Static Text		هو عبارة عن حقل يقوم بعرض نص ثابت، يستخدم بهدف توضيح عمل عنصر من عناصر الواجهة مثلاً.
Pop-up Menu		هي عبارة عن قائمة منسدلة عند الضغط على السهم الموجود ضمنها تفتح وتعرض للمستخدم عدد من الخيارات، مثلاً نريد رسم مجموعة من التوابع نضع ضمنها الخيارات يضغط المستخدم لاختيار التابع المراد رسمه.
List Box		تقوم هذه المجموعة بعرض قائمة من العناصر وتتيح للمستخدم باختيار عنصر أو أكثر.
Table		تستخدم من اجل إنشاء جدول من العناصر.
Axes		وجود هذا العنصر ضمن الواجهة بيانية التخاطبية GUI يتيح عملية إظهار بيانيات graphics مثلاً منحني تابع ما أو صورة. يمكن التحكم بخصائص هذا العنصر بشكل كبير للتحكم بالإظهار.
Toolbar		يتيح لك هذا العنصر بإنشاء شريط أدوات خاص بالواجهة أو التطبيق الذي تعمل عليه من أجل إغنائه بخيارات إضافية.

بعد هذه المقدمة النظرية عن العناصر، سنقوم برؤية شكل معظم العناصر ضمن واجهة إضافةً لتجريب عملها، هذه الواجهة عبارة عن مثال تم تنجيزه مسبقاً ضمن MATLAB، لتشغيله قم بكتابة الرماز التالي:

```
copyfile(fullfile(docroot, 'techdoc','creating_guis',...
    'examples','controlsuite*.*')),...
    fileattrib('controlsuite*.*', '+w');
guide controlsuite.fig
```

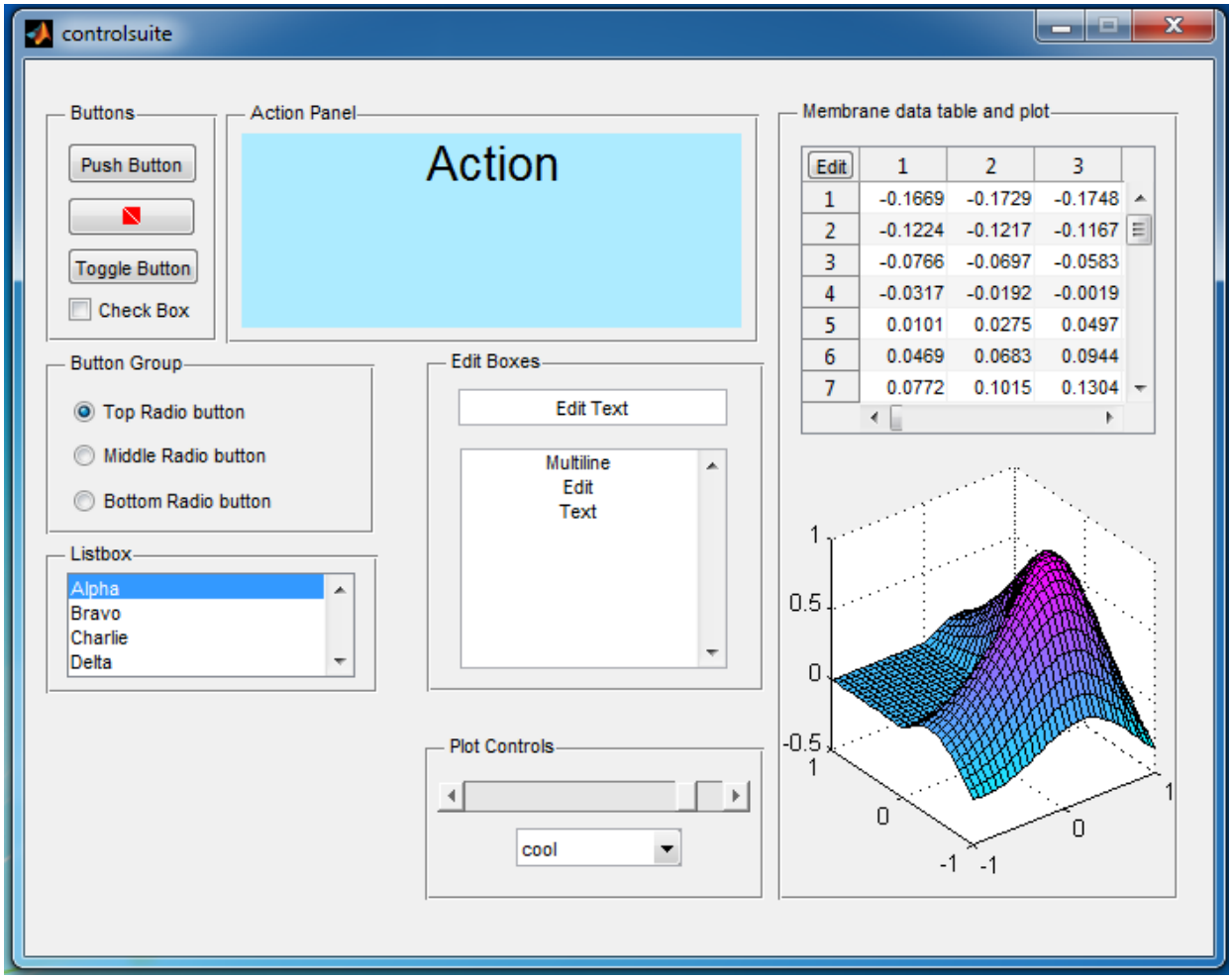
ملاحظة: يجب التأكد من مسار العمل تجنباً للأخطاء.

عند تنفيذ الرماز السابق ستفتح واجهة كما في الشكل التالي:



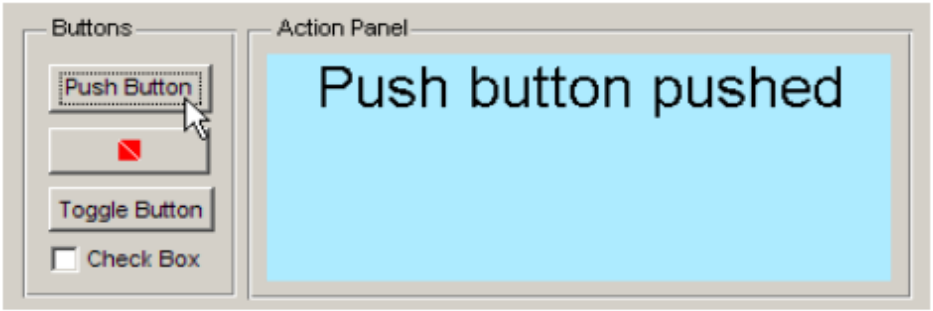
من أجل تشغيل الواجهة، ضمن الشريط العلوي اضغط على tools ومن ثم اختر Run أو اضغط على أيقونة Run مباشرةً كما يوضح الشكل.

عند التشغيل تحصل على الشكل التالي:



من أجل معاينة الرماز البرمجي المنجز وتوابع CallBacks يمكن من View اختيار Editor سيتم فتح ملف m. يحتوي على الرماز، أو يمكن من خلال الأيقونة الخاصة ب Editor، هذه الخيارات موضحة في الشكل.

قم بالتعرف على الواجهة أكثر اضغط على الأزرار ولاحظ ماذا يتغير ضمن الواجهة، يوجد Action Panel تقوم بعرض للمستخدم الحدث الذي قام بفعله، مثلاً عند الضغط على Push Button تجد:



التعرف أكثر على GUIDE واستخدامه في بناء واجهة بيانية تخاطبية Getting to know more about GUIDE and usre it to build a GUI

قبل التعرف بشكل أكبر على GUIDE والبدء ببناء مثال توضيحي، سنعرض مراحل بناء واجهة بيانية تخاطبية GUI لتطبيق أو برنامج ما ضمن MATLAB، هذه المراحل هي:

1. تصميم واجهة البرنامج أو التطبيق.
2. ضبط الخصائص.
3. برمجة توابع Callbacks.

1. تصميم واجهة البرنامج بشكل عام يتم عبر وضع العناصر المرغوبة في المكان المناسب، يتطلب ذلك تفكير أولي ورسم للشكل النهائي الذي يرغب المستخدم في الحصول عليه قبل البدء بالتصميم، إن استخدام العناصر الجاهزة ضمن GUIDE والتي تم شرح لعملها مسبقاً توفر على المستخدم الوقت من حيث البداية في كتابة الرماز البرمجي المطلوب من الصفر ،إذا تومن له، عموماً، العديد من إمكانيات الإظهار والعمليات البرمجية المختلفة ، يتم توضيع العناصر على النافذة الرئيسية (figure) إما بالنقر المزدوج عليها في صندوق الأدوات حيث تتوضع في الزاوية اليسرى ومن ثم تحريكها ضمن النافذة إلى المكان المرغوب (عن طريق السحب والإفلات) وتغيير حجمها عن طريق حوافها ، أو بالنقر على العنصر (مرة واحدة) ومن ثم تشكيها (رسمها) ضمن النافذة بالحجم والمكان المطلوبين.
2. لكل عنصر من العناصر، بما في ذلك نافذة البرنامج، مجموعة محددة من الخصائص تحدد مظهر وسلوك العنصر، وعندما وضع عنصر ما ضمن الواجهة يضع MATLAB قيماً افتراضية لخصائصها، ويمكن ضبط هذه الخصائص إما أثناء التصميم أو أثناء التنفيذ. واحدة من أهم الخصائص هي اسم العنصر، ينصح بتسمية العناصر بأسماء واضحة تعبر عن وظيفتها، وذلك لبناء رماز برمجي فعال من السهل العودة إليه، كما انه مع زيادة عدد العناصر أو العودة للواجهة لاحقاً لا يعود بالإمكانية تذكر اسم زر معين، مثلاً عند وضع أربعة ازرار من نوع Push Button تكون الأسماء Push_Button_1، Push_Button_2، Push_Button_3، Push_Button_4 وهنا احتمال الخطأ كبير لذلك مثلاً لنفترض أننا نريد الاستفادة من هذا الأزرار من أجل تنفيذ العمليات التالية جمع، طرح، ضرب وقسمة عندها ينصح بتسمية الأزرار بالأسماء التالية:myAdd, mySub, myMult, myDiv.
3. أوضحنا في فقرات سابقة أن كل حدث ينتج عن فعل من المستخدم يجب أن يولد استجابة من الواجهة البيانية تتم هذه الاستجابة عن طريق توابع CallBacks والتي يكتب ضمنها رماز برمجي لتوصيف عمل الزر، المحور ، ... أو أي عنصر آخر.

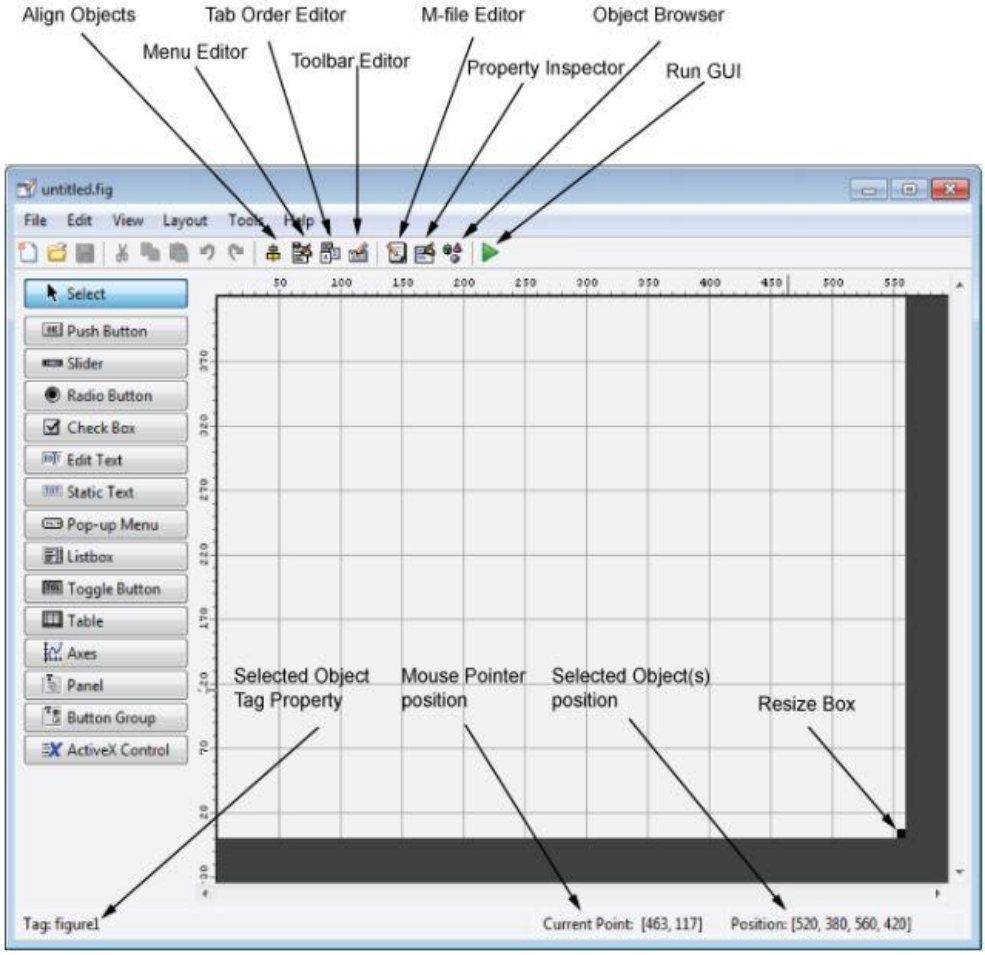
يقوم المصمم أو المبرمج باستخدام لغة MATLAB بإعداد تابع callback لكل حدث، يحدد هذا التابع شكل الاستجابة لهذا الحدث، يتم الوصول إلى ملف البرمجة m. الخاص بالواجهة عن طريق النقر بالزر اليميني على

العنصر ضمنالواجهة واختيار view callbacks ومن ثم اختيار callbacks وهنا يتم الانتقال إلى القسم البرمجي داخل ملف m. وتبدأ عملية برمجة التابع.

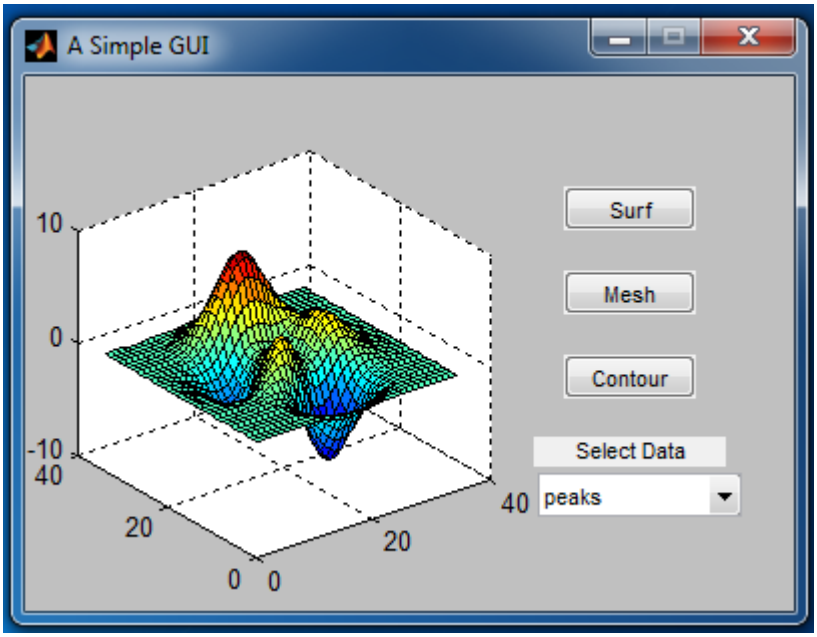
في واجهات MATLAB يتميز كل عنصر واجهة عن أي متحول بواسطة المقبض البرمجي handles وباسم برمجي يدعى tag وهو موجود ضمن خصائصه ويمكن تغييره ولكن يجب الانتباه إلى استخدام الاسم الجديد أثناء البرمجة.

التعامل بشكل أساسي مع العناصر يتم عبر التوابع set وget ، set يستخدم لوضع قيمة ما أو فعل ما ضمن العنصر مثلاً لوضع نتيجة الجمع ضمن حقل text box يعبر عن الخرج، أما get يستخدم للحصول على قيمة ما أو حدث ما، مثلاً الحصول على الدخل من من مربع نصي edit text.

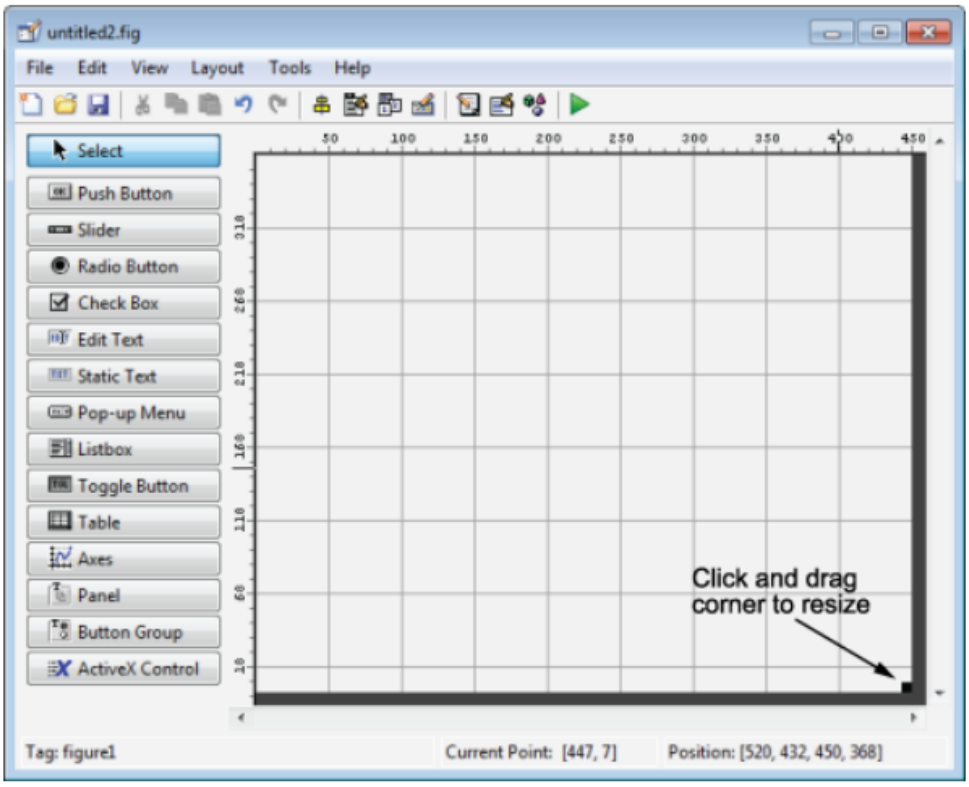
لنبدأ التعرف أكثر على GUIDE بعد عرض المراحل الخاصة ببناء الواجهات، يبين الشكل الخيارات المتاحة من نافذة GUIDE:



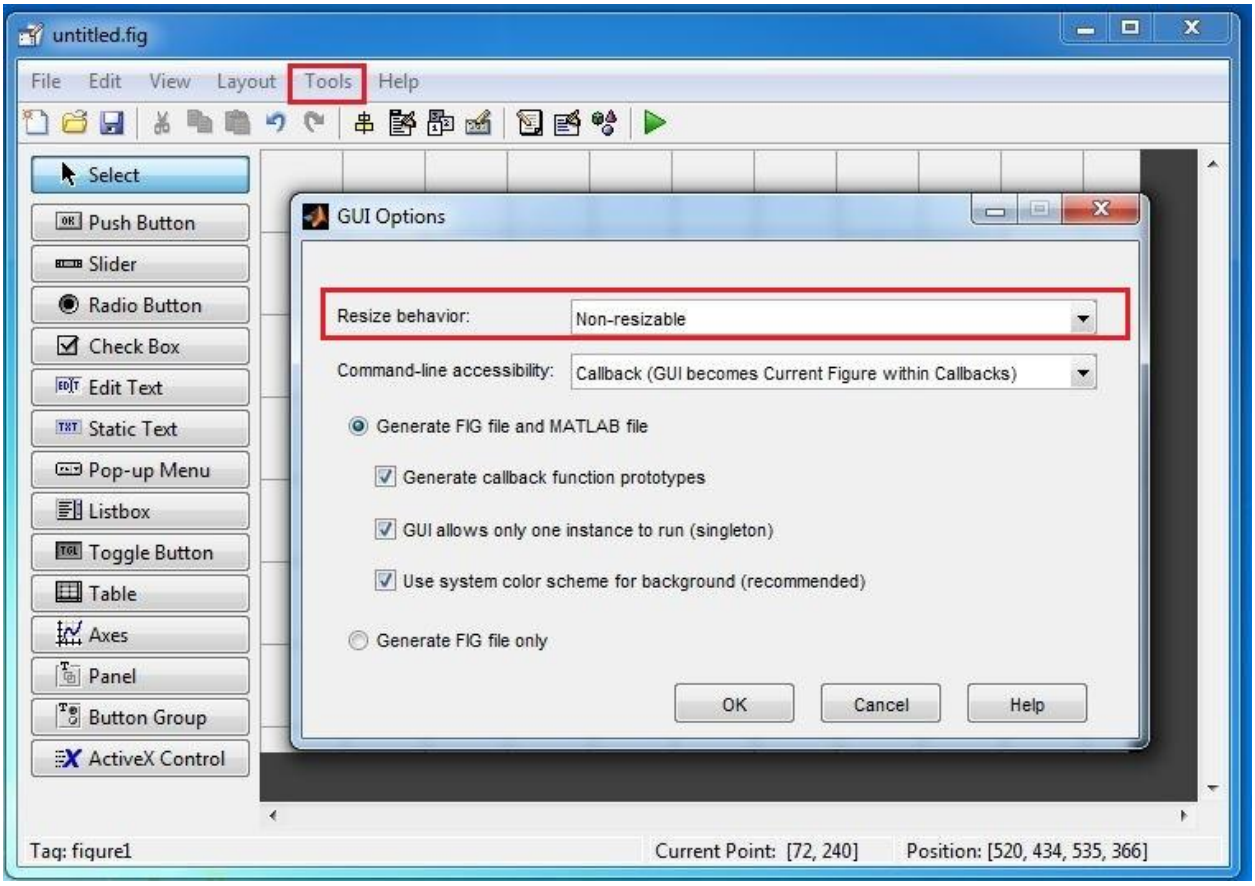
سنبدأ بإنشاء واجهة بيانية تخاطبية تسمح للمستخدم باختيار نوع من المعطيات من ضمن ثلاثة مجموعات من المعطيات ومن ثم عرضها بأحد الأشكال الثلاثة المتاحة ضمن الواجهة، الشكل النهائي لها هو:



بعد فتح واجهة بيانية جديدة، يمكن تحديد حجم الواجهة النهائية بشكل أولي عن طريق الزاوية اليمنى السفلية يوجد مربع أسود اللون يتم عبر تحريكه التحكم بالحجم.



بعد تحديد الحجم في الواجهة الفارغة السابقة سيتم، عند التنفيذ، الحصول على واجهة بيانية بنفس الحجم الذي جرى تحديده كما أنها ستكون غير قابلة لتغيير حجمها ، يمكن أيضاً التحكم بالقدرة على إعادة تعيين الحجم عن طريق شريط الأدوات اختر من Tools الخيار GUI Options ستظهر أمامك الواجهة التالية يمكن تغيير خيار Resize Behavior:



سنبدأ بتنسيق الشكل الخارجي (المظهر) للواجهة البيانية أي تصميم الواجهة إضافةً لضبط خصائص الأزرار.

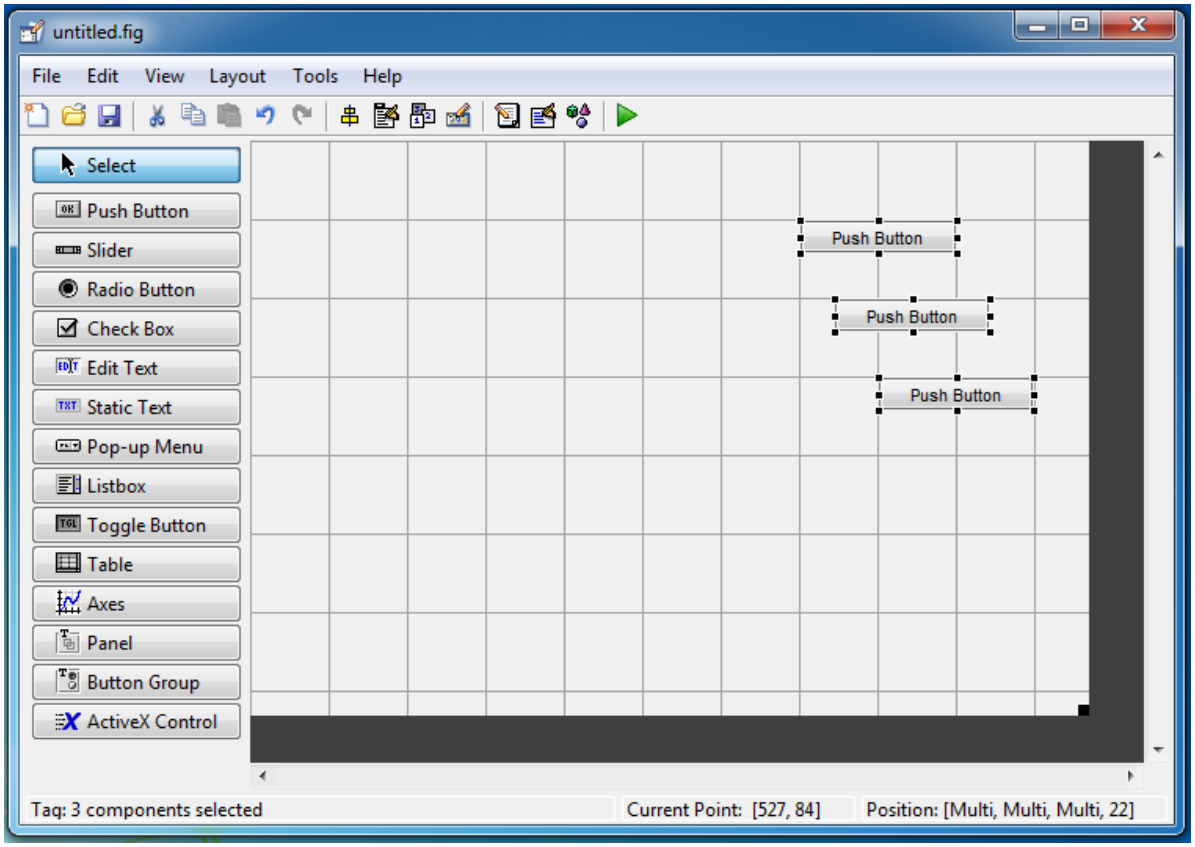
سيتم إضافة، إزاحة ومحاذاة Align، والإشارة label إلى العناصر المكونة للواجهة التخاطبية.

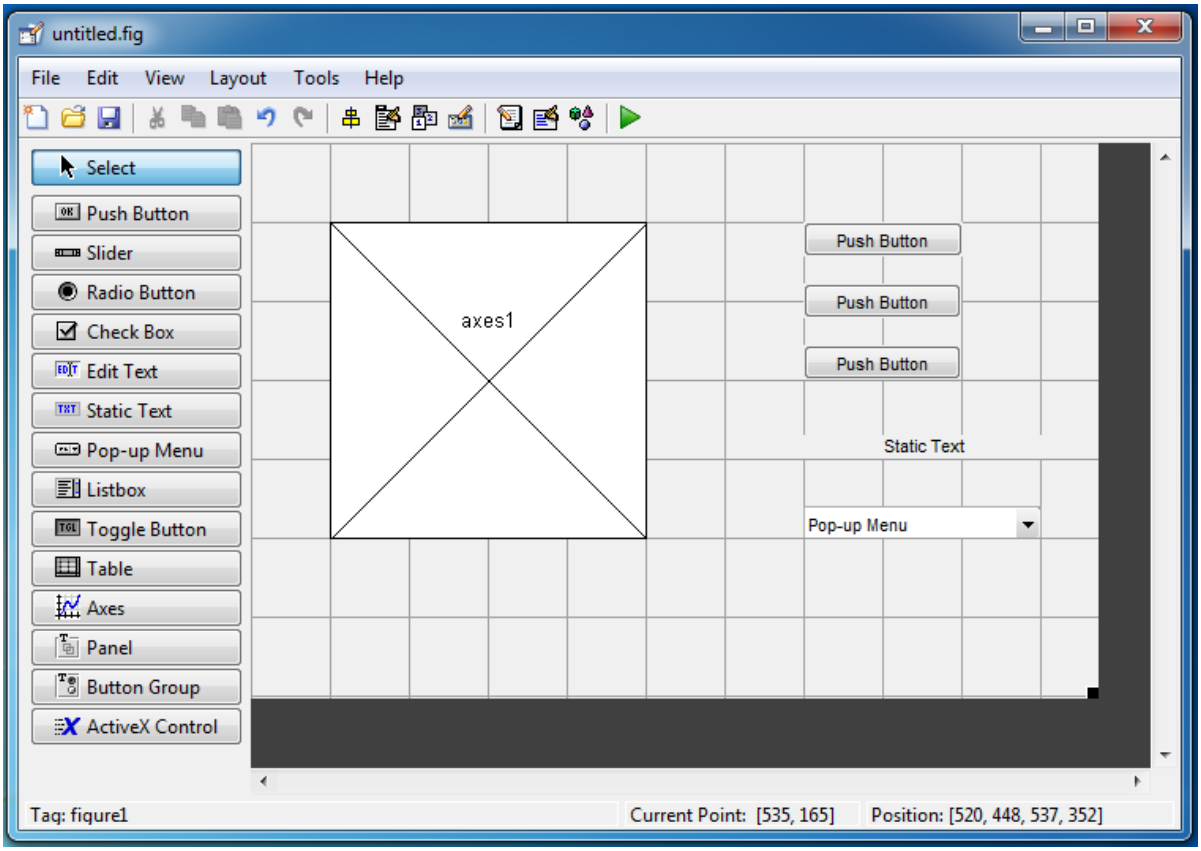
1. قم بإضافة ثلاثة أزرار من نوع Push Button لتصبح الواجهة كما في الشكل التالي:

بعد ذلك قم بإضافة باقي العناصر المكونة للواجهة البيانية وهي:

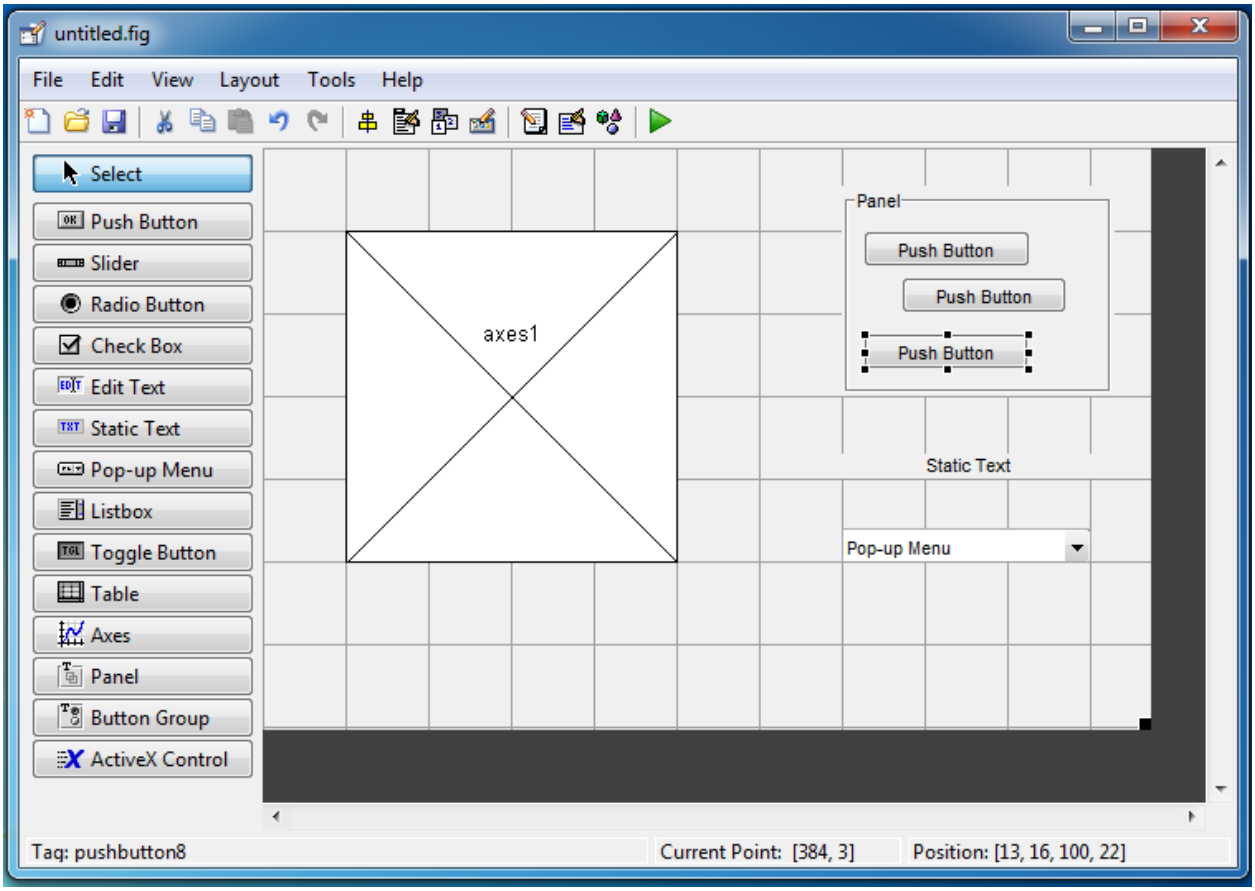
- Static text
- Pop-up menu
- Axes

تصبح الواجهة على الشكل التالي

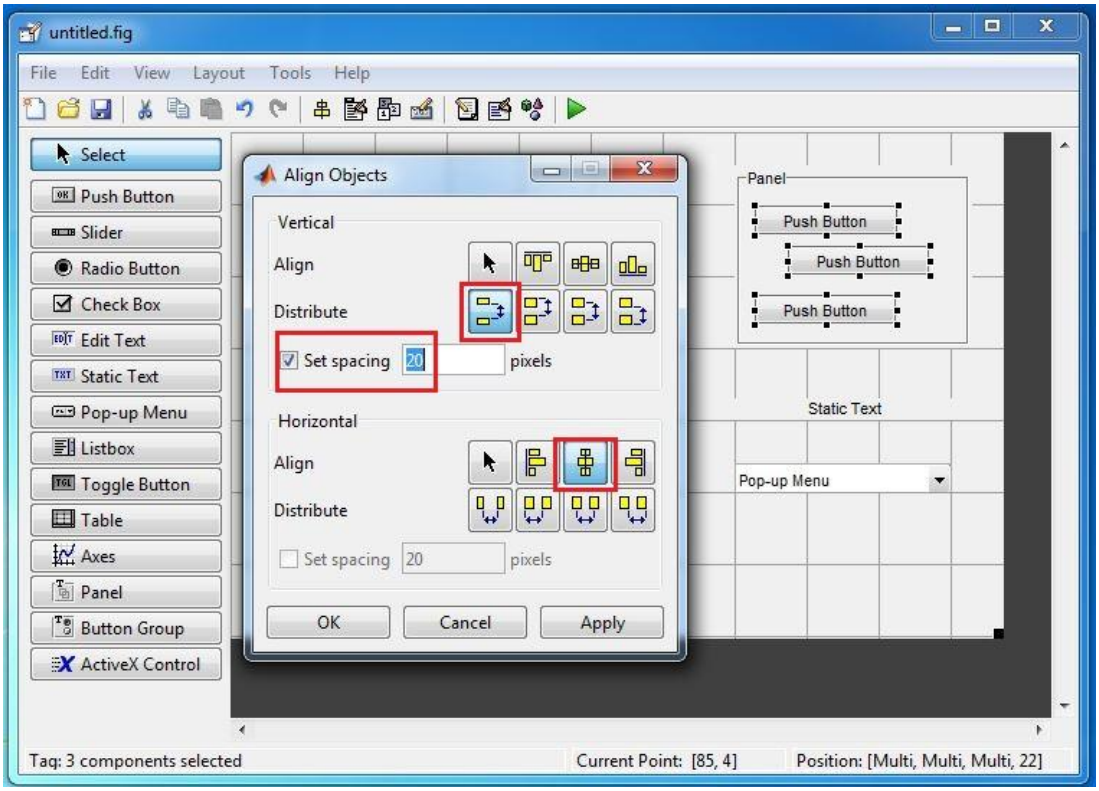




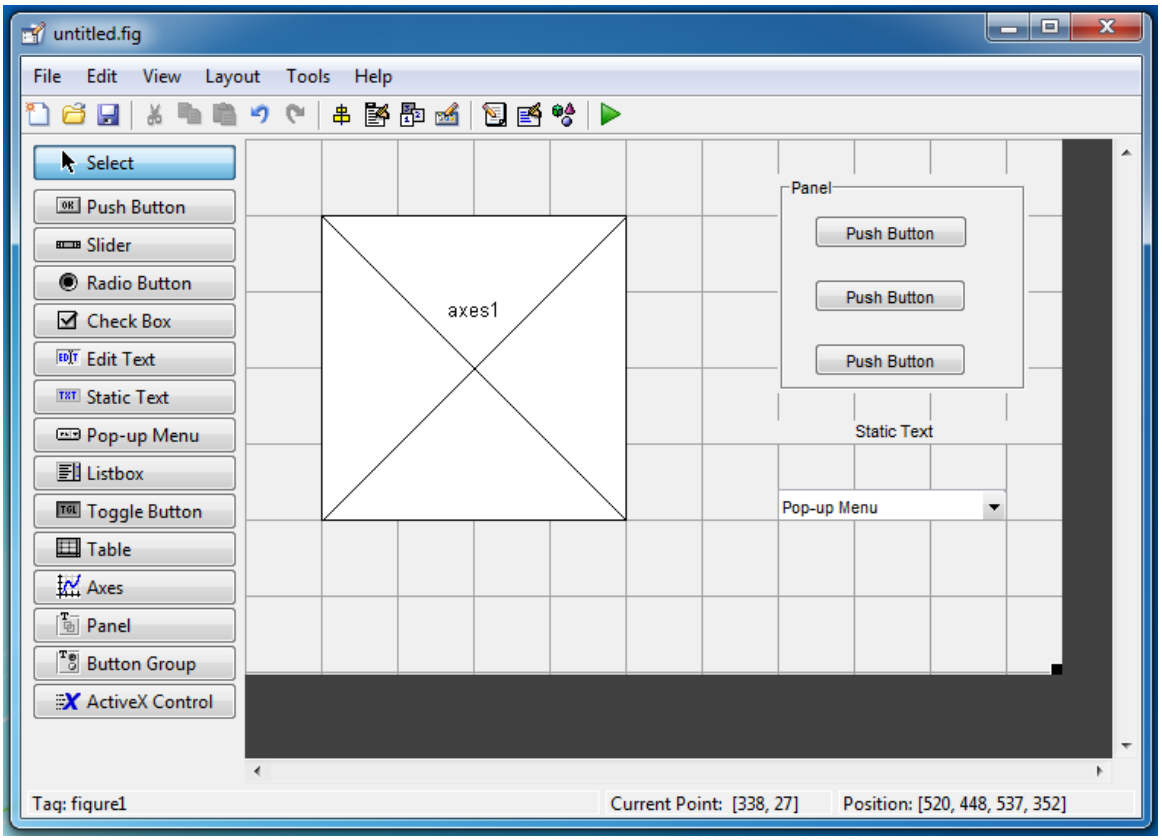
من أجل تنظيم أزرار Push Button بشكل أوضح وأكثر رتابة يمكن إحضار Panel ووضع الأزرار ضمنها كما في الشكل:



يمكن الاستفادة من أداة المحاذاة للتحكم بتموضع عناصر لها نفس الأب أو الجذر مثلاً مجموعة من الأزرار ضمن Panel، قم بتحديد الأزرار Push button الثلاثة عن طريق المحافظة على الضغط على Ctrl والضغط عليهم جميعاً ومن ثم قم باختيار Tools → Align Object وقم باختيار الإعدادات التالية موضحة في الشكل ومن ثم اضغط على OK.



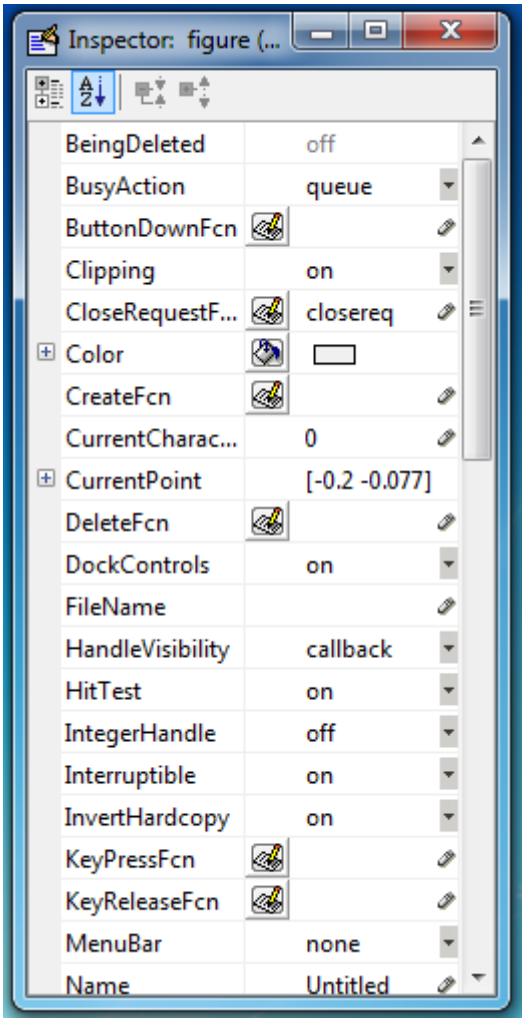
تحصل على الشكل التالي:



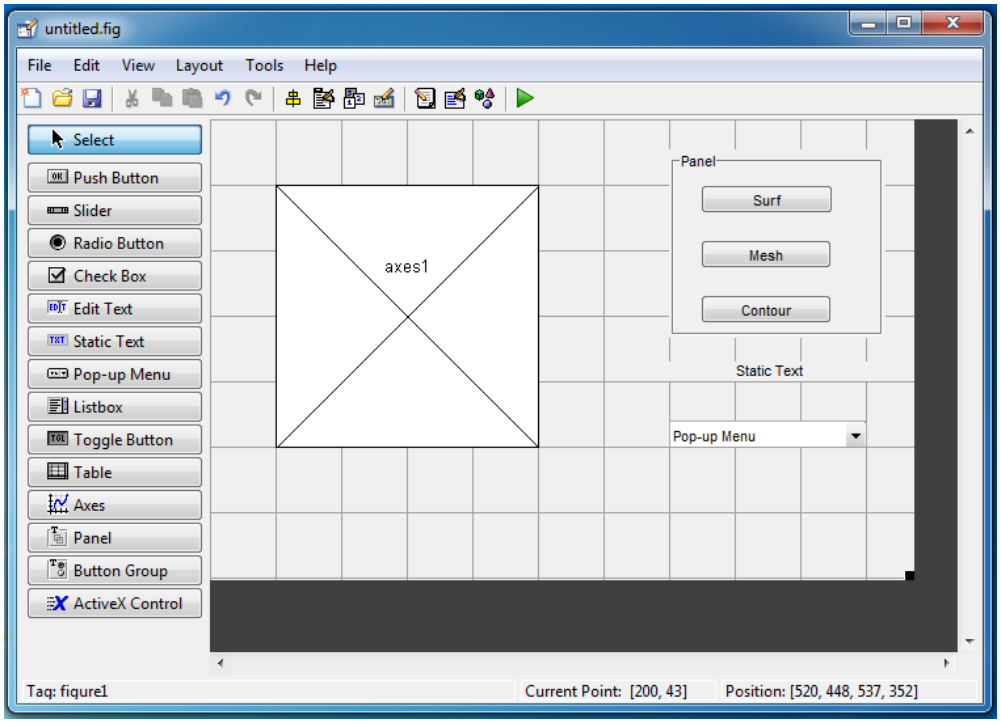
ويمكن التحكم بتموضع الأزرار، والحقول النصية، والقوائم، والمحاور بالشكل الذي يرغب به المستخدم.

بعد ذلك سنقوم بالإشارة Label إلى الازرار أو إعادة تسميتها وذلك بهدف تسهيل الوصول إليها عند الرغبة ببرمجتها وإضافة أسماء زاضحة لها تعبر عن عملها.

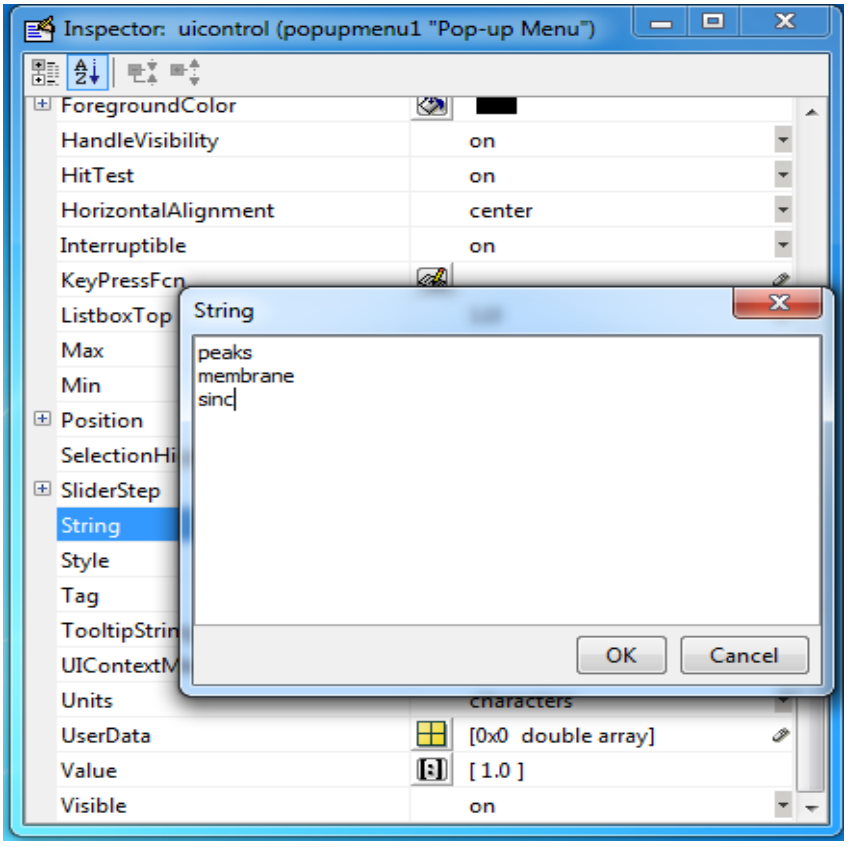
كل واحد من الأزرار Push Button سيقوم بوظيفة محدّدة وهي رسم إحدى الرسمات ثلاثية الأبعاد وهي surf, mesh, contour، يمكن اختيار من View → Property Inspector. ستفتح الواجهة التالية:



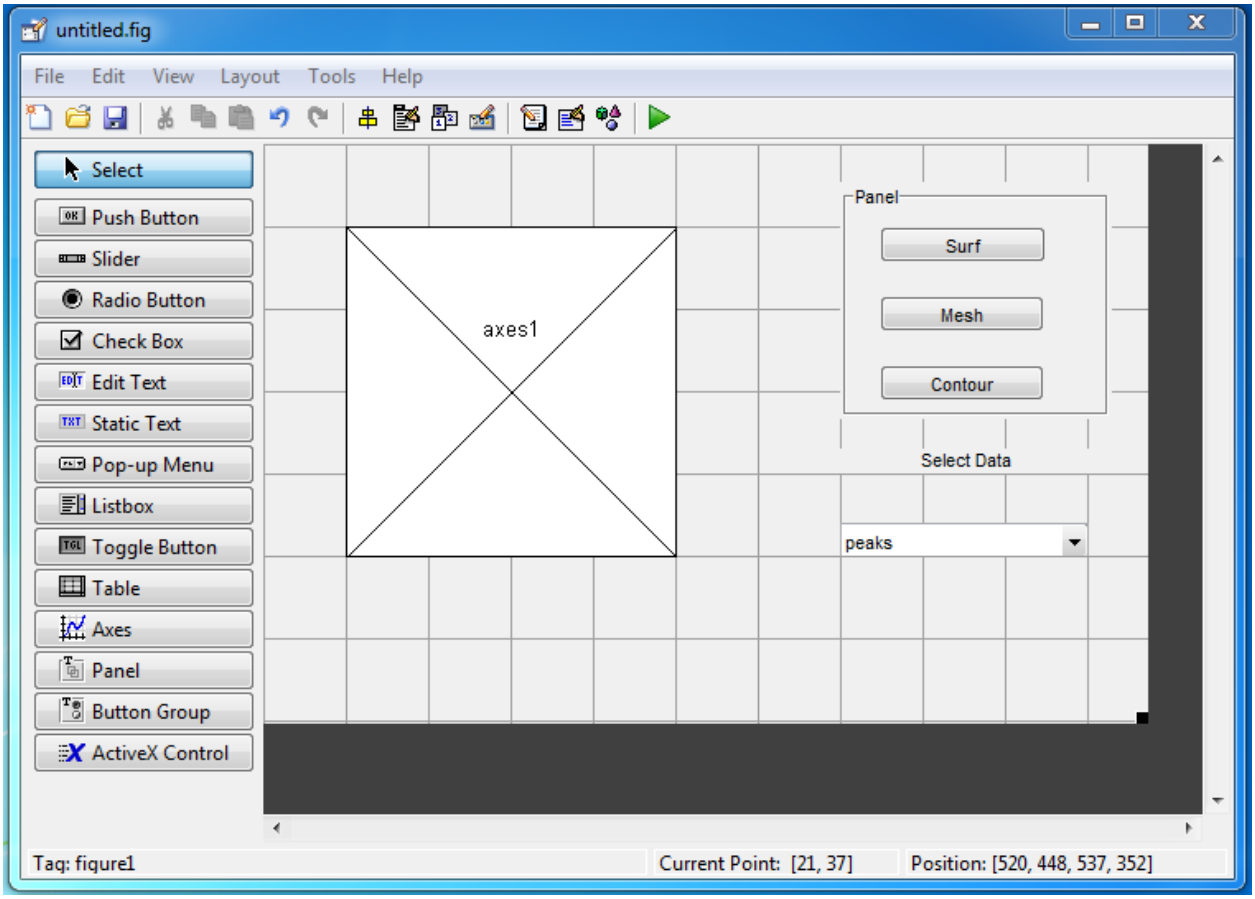
ابحث عن string من اجل اول زر وقم بتغيير الاسم الذي يتم عرضه إلى surf ثم للزر الثاني اكتب Mesh ثم contour لتحصل على الشكل التالي للواجهة:



أعد العملية السابقة ل static text واكتب له اسم هو Select Data، و ل pop-up menu واكتب ضمن string الكلمات التالية:



تصبح الواجهة على الشكل:



ابحث عن tag من أجل أول زر ضمن Property Inspector قم بتغيير tag للعنصر إلى الاسم التالي:

surf_pushbutton_Callback

ومن ثم أعد العملية للزرين الباقيين

mesh_pushbutton_Callback

contour_pushbutton_Callback

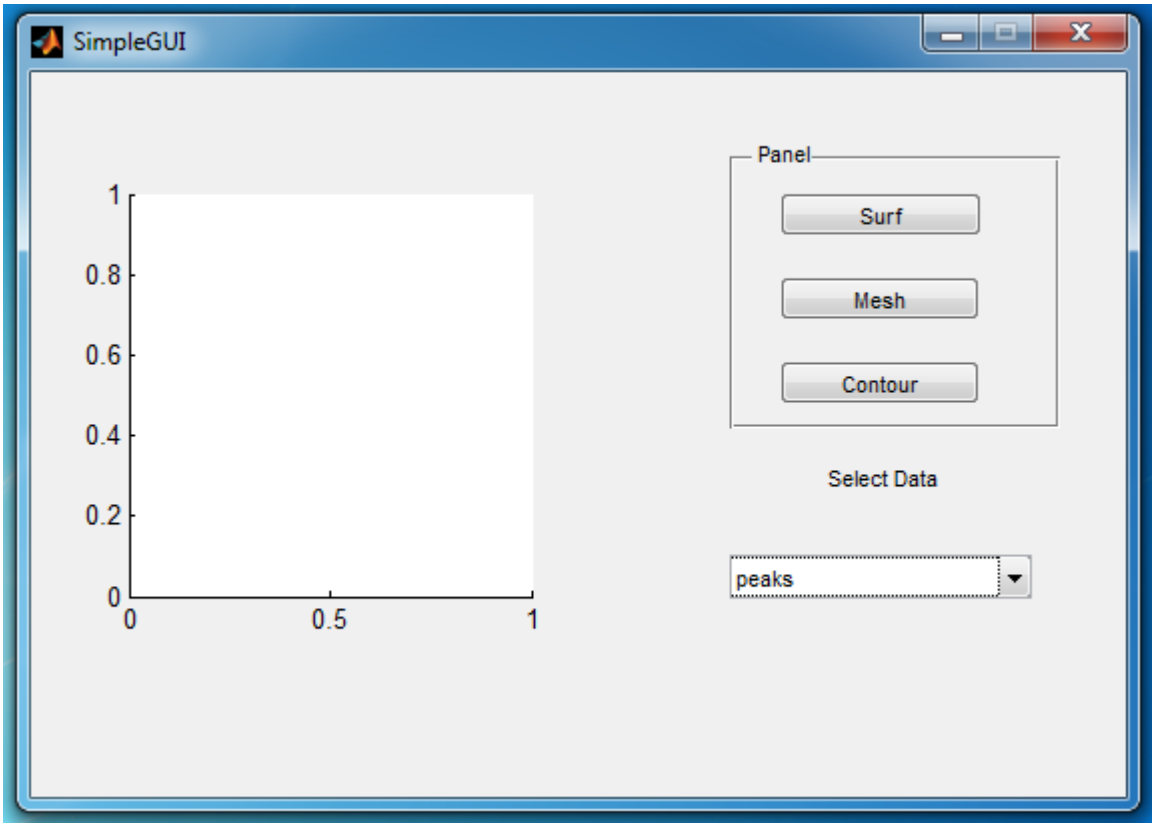
و من أجل pop-up menu

plot_popup_Callback

بعد ذلك قم بحفظ الواجهة باسم SimpleGUI، بعد الحفظ يم توليد ملفين الأول fig. والذي يحتوي على وصف للشكل الخارجي للواجهة وملف m. يحتوي على الرماز البرمجي الذي يتضمن التوابع التي تتحكم بسلوك الواجهة البيانية.

ستلاحظ تلقائياً فتح الرماز البرمجي التي تم توليده تلقائياً. يرجى عدم حذف أي سطر من السطور البرمجية التي تم توليدها.

قم بتشغيل الواجهة عن طريق زر Run تحصل على الشكل التالي:



سننتقل الآن إلى البرمجة، نلاحظ إزا قمنا بفتح الرماز البرمجي المتولد وجود توابع عديدة إلا أن الواجهة لا تقوم بأي عمل وذلك يرجع لأنه تم فقط إنشاء التوابع دون كتابة تعليمات توصف السلوك عند وقوع حدث ما.

مثلاً نلاظ التوابع التالية:

```
function plot_popup_Callback_Callback(hObject, eventdata, handles)

function surf_pushbutton_Callback_Callback(hObject, eventdata, handles)

function mesh_pushbutton_Callback_Callback(hObject, eventdata, handles)

function contour_pushbutton_Callback_Callback(hObject, eventdata, handles)
```

إضافةً لتوابع callbacks السابقة نلاحظ إنشاء بعض التوابع مثل:

```
function SimpleGUI_OpeningFcn(hObject, eventdata, handles, varargin)


function plot_popup_Callback_CreateFcn(hObject, eventdata, handles)
```

إن توابع CreateFcn و OpeningFcn توصف لنا ما العمل الذي يقوم به الزر أو المحور أو .... عند تشغيل الواجهة.

لاحظنا عند تشغيل الواجهة لايوجد أي معطيات للعرض لذلك سنقوم الآن فقط ببرمجة ماذا يجب أن يتم عرضه على المحاور عند تشغيل الواجهة دون الخوض بتفاصيل عمل الأزرار.

يمكن البحث ضمن ملف .m المتولد عن التابع

SimpleGUI_OpeningFcn

أو يمكن ضمن EDITOR يوجد قسم للبحث هو Navigator اختر GO TO

SimpleGUI_OpeningFcn

نلاحظ الانتقال إلى التابع الذي يحتوي على السطور البرمجية التالية:

```
function SimpleGUI_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to SimpleGUI (see VARARGIN)

% Choose default command line output for SimpleGUI
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes SimpleGUI wait for user response (see UIRESUME)
% uiwait(handles.figure1);
```

سنقوم بإنشاء المعطيات من أجل رسمها باستخدام الرماز التالي:

بعد السطر البرمجي التالي

```
% varargin   command line arguments to SimpleGUI (see VARARGIN)
```

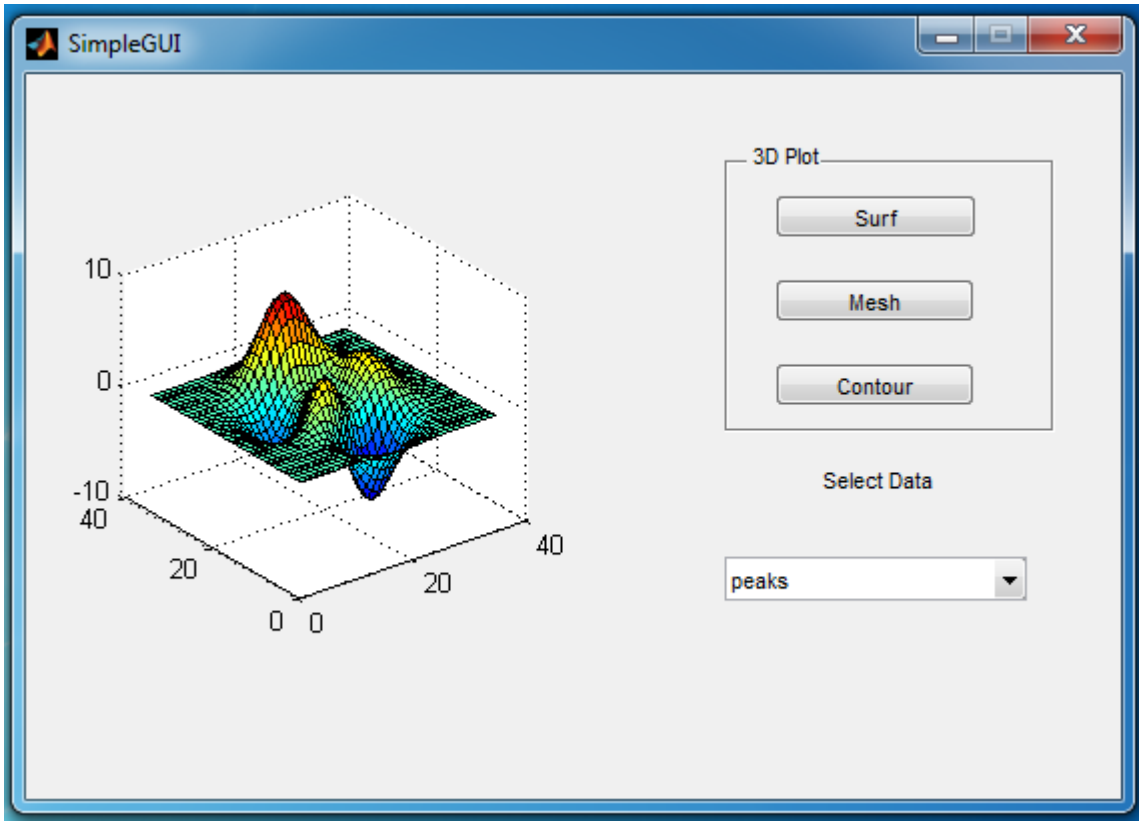
اكتب ما يلي:

```
handles.peaks=peaks(35);
handles.membrane=membrane;
[x,y] = meshgrid(-8:.5:8);
r = sqrt(x.^2+y.^2) + eps;
```

```
sinc = sin(r)./r;
handles.sinc = sinc;
% Set the current data value.
handles.current_data = handles.peaks;
surf(handles.current_data)
```

الأسطر الستة الأولى من الرماز السابق تقوم بتوليد المعطيات باستخدام التوابع المعرفة ضمن MATLAB وهي peaks, membrane and sinc، يتم حفظ المعطيات ضمن البنية handles، هذا الوسيط argument هو متاح لجميع توابع callbacks أي يمكن استدعائه ضمن جسم أي تابع callbacks والحصول على المعطيات المخزنة ضمنه.

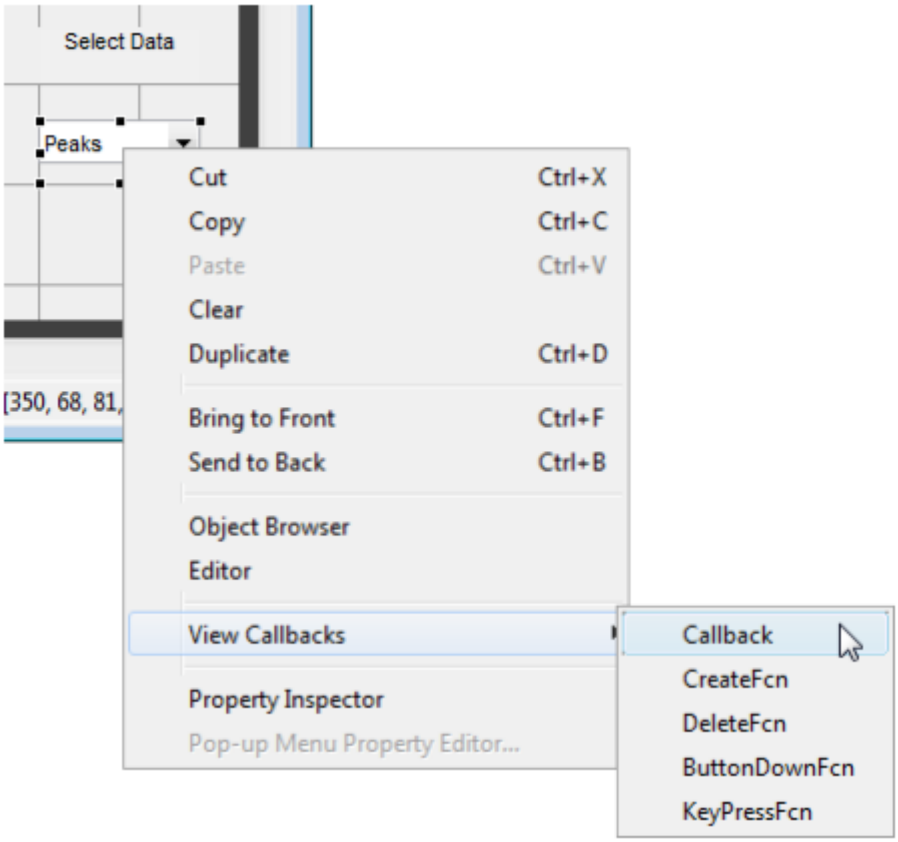
السطرين الأخيرين تقوم بإنشاء المعطيات الحالية current data وتسند لها peaks ومن ثم تعرض المنحني surf من أجل المعطيات peaks، عند تشغيل الواجهة سنحصل على الشكل التالي:



لنقم الآن ببرمجة عمل pop-up menu.

تعرض لنا القائمة المنسدلة pop-up menu في هذا المثال خيارات المعطيات التي يرغب المستخدم في رسمها. عندما يختار المستخدم أحد خيارات الرسم الثلاث contour, ,mesh, surf، تقوم برمجية MATLAB بوضع (set) القيمة (VALUE) التي تؤشر على السلسلة المحرفية التي تم اختيارها، هذه القيمة يمكن قرائتها ضمن Callbacks من أجل تحديد نمط المعطيات الواجب عرضها ومن ثم يتم وضع المعطيات الواجب عرضها ضمن handles.current_data.

من أجل الدخول إلى callback الخاصة بـ pop-up menu اضغط بالزر اليميني على العنصر ومن ثم اضغط Callback → View Callbacks كما في الشكل:



يتم عرض الرماز البرمجي التالي

```
% --- Executes on selection change in plot_popup_Callback.
function plot_popup_Callback_Callback(hObject, eventdata, handles)
% hObject    handle to plot_popup_Callback (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
```

ومن ثم قم يكتابة الرماز التالي:

```
% Determine the selected data set.
str = get(hObject, 'String');
val = get(hObject,'Value');
% Set current data to the selected data set.
switch str{val};
case 'peaks' % User selects peaks.
```

```
handles.current_data = handles.peaks;
case 'membrane' % User selects membrane.
handles.current_data = handles.membrane;
case 'sinc' % User selects sinc.
handles.current_data = handles.sinc;
end
% Save the handles structure.
guidata(hObject,handles)
```

يبين لنا الرماز السابق خاصتين مهمتين من خواص القائمة المنسدلة Pop-up menu وهي القيم المعادة من هذه القائمة وهي نوعان:

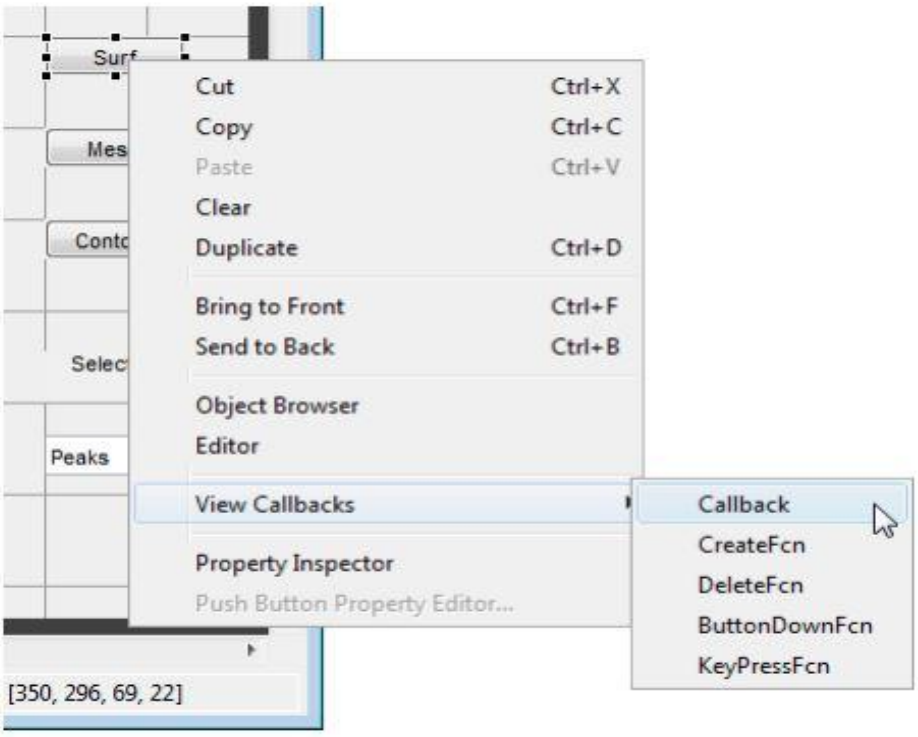
- String عبارة عن صفيفة خلية Cell Array تحتوي على محتوى القائمة.
- Value يحتوي على الدليل index للسطر المختار (نوع المعطيات المختار) داخل محتوى القائمة.

ثم بعد ذلك نستخدم تعليمة switch من أجل إسناد المعطيات المختارة إلى المعطيات الحالية cureent_data وذلك حسب القيمة التي جرى اختيارها.

لنقم الآن ببرمجة عمل push button.

كل زر من هذه الأزرار يقوم برسم نوع مختلف من أنواع الرسوم Plots وذلك على أحد انماط المعطيات التي جرى اختيارها من قبل المستخدم، تقوم callbacks الخاص ب Push button بالحصول على المعطيات من handles ومن ثم ترسم هذه المعطيات على المحور.

من أجل الدخول إلى callback الخاصة بـ Surf اضغط بالزر اليميني على العنصر ومن ثم اضغط View Callbacks → Callback كما في الشكل:



يتم عرض الرماز البرمجي التالي:

```
% --- Executes on button press in surf_pushbutton_Callback.
function surf_pushbutton_Callback_Callback(hObject, eventdata, handles)
% hObject    handle to surf_pushbutton_Callback (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
```

ومن ثم قم يكتابة الرماز التالي:

```
% Display surf plot of the currently selected data.
surf(handles.current_data);
```

قم بعادة العملية السابقة لكل من الأزرار mesh و contour واكتب الرمازات التالية:

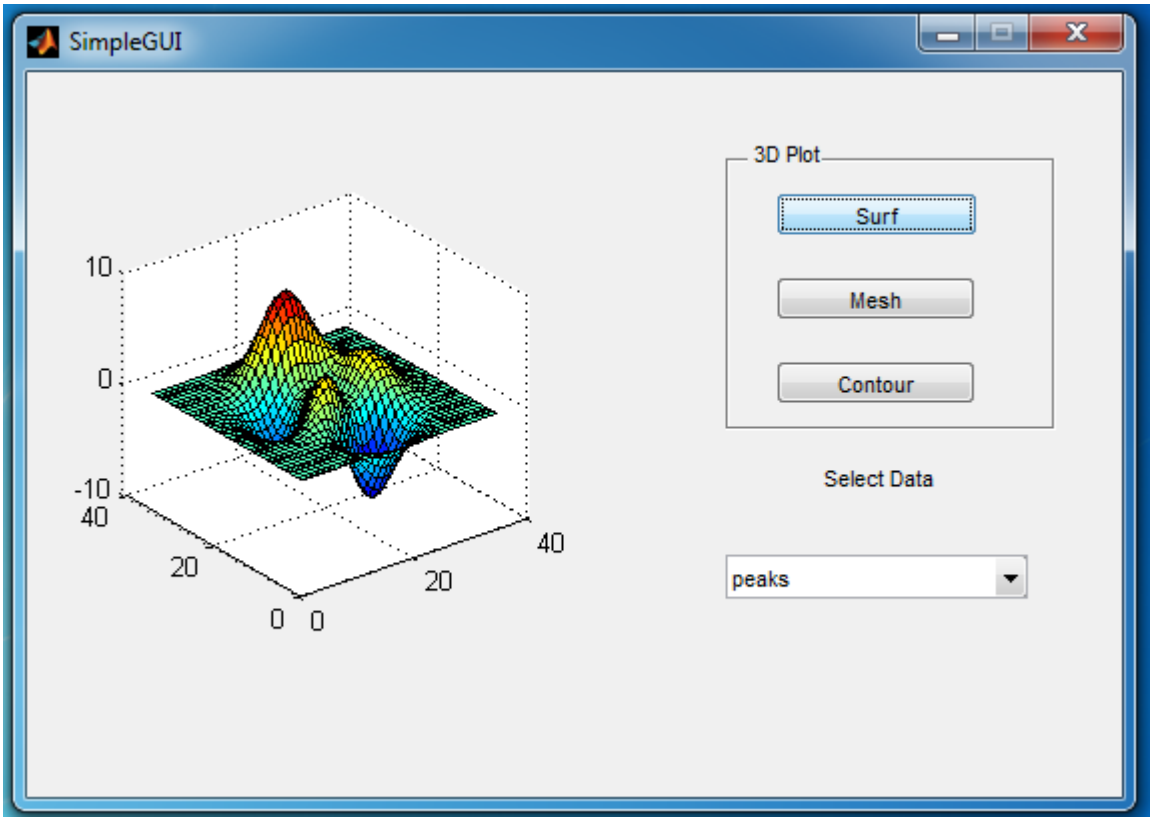
من أجل زر mesh

```
% Display mesh plot of the currently selected data.
mesh(handles.current_data);
```

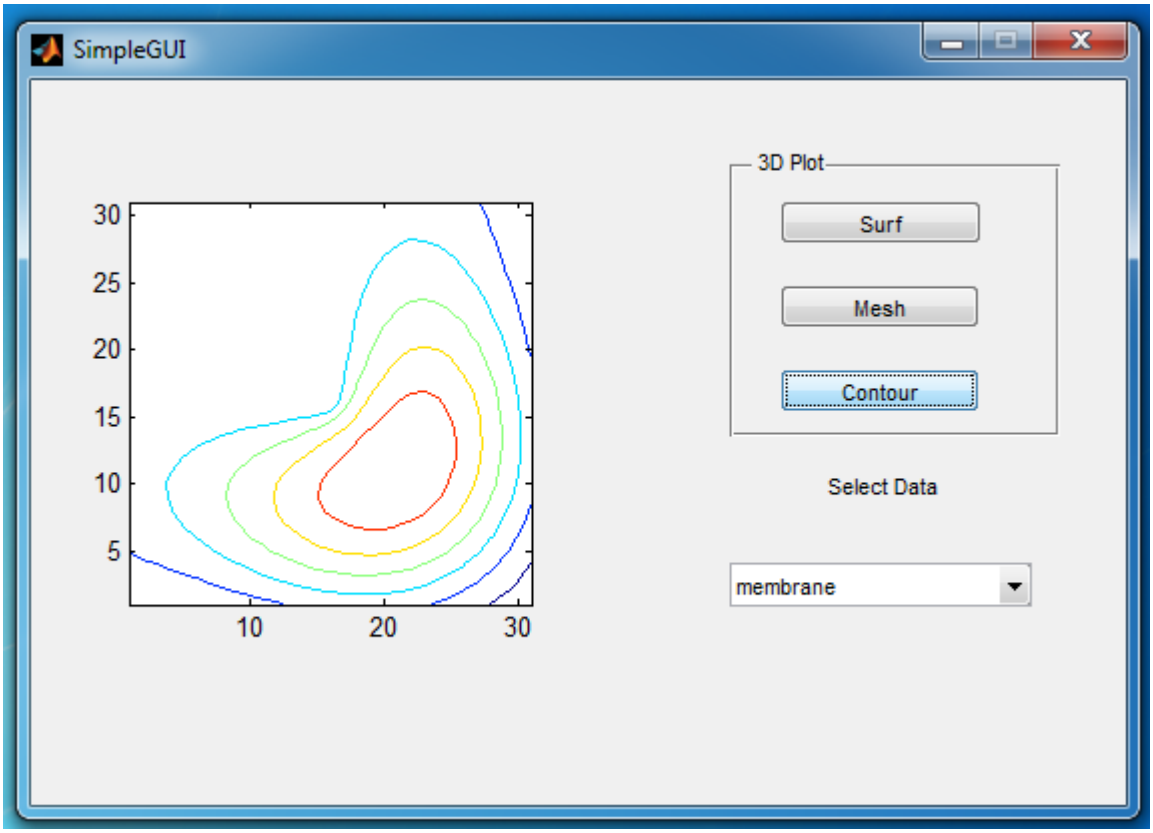
من أجل زر contour

```
% Display contour plot of the currently selected data.
contour(handles.current_data);
```

قم بعد ذلك بحفظ الرماز عن طريق File → Save. وبعدها قم بتشغيل الواجهة ستحصل على الشكل التالي



قم باختيار نوعي المعطيات الباقيين وقم بتجريب تغيير الرسم الذي تود بإظهاره للتاكد من عمل الواجهة. مثلاً: اختيار membrane مع contour تحصل على



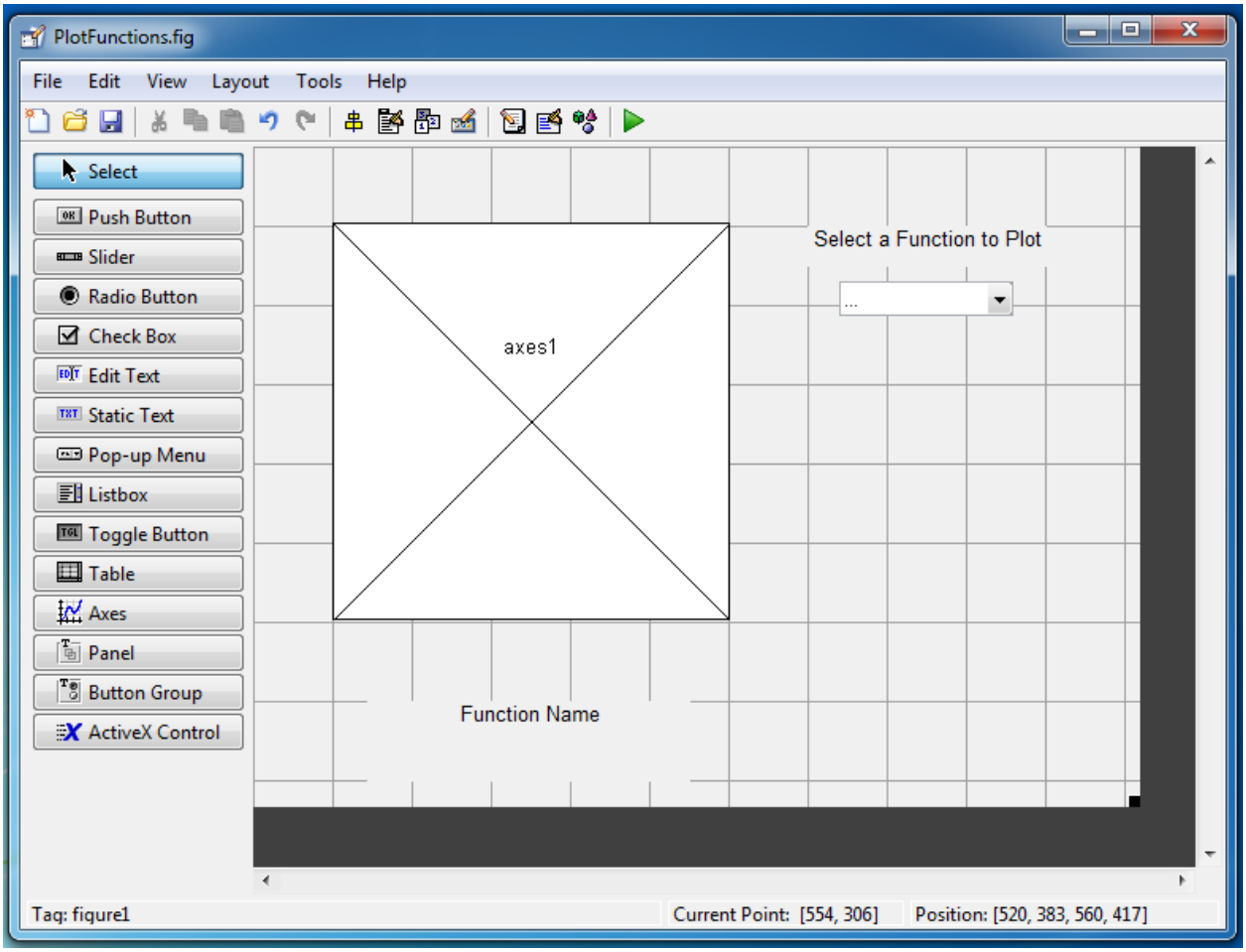
أمثلة Examples

مثال 1: ضمن هذا المثال سنقوم بإنشاء واجهة بيانية تسمح للمستخدم باختيار أحد التاوبع sin, cos, exp ومن ثم رسم منحني التابع على مجال زمني ثابت هو $[0 - 2\pi]$ وبخطوة مقدارها $\frac{\pi}{100}$.

نحتاج إلى العناصر التالية:

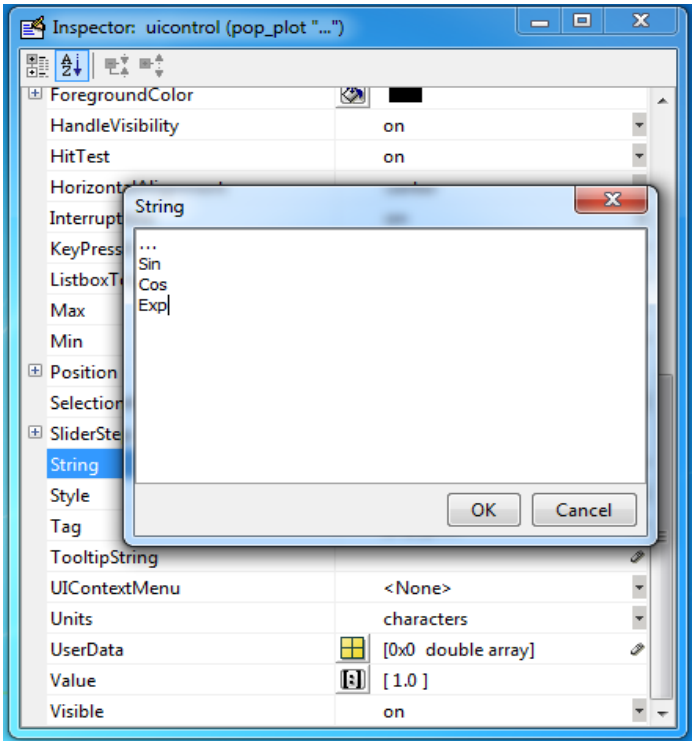
- Pop-up menu قائمة منسدلة لاختيار التابع الذي سيتم رسم المنحني المعبر عنه.
- Static Text نص ثابت يوضع فوق القائمة المندسلة يحتوي على عبارة توضيحية لعمل القائمة حيث سنكتب "Select a Function to Plot".
- عنصر Axes من أجل رسم منحني التابع ضمنه.
- Static Text يعرض للمستخدم التابع الذي تم اختياره.

الشكل الأولي للواجهة هو:

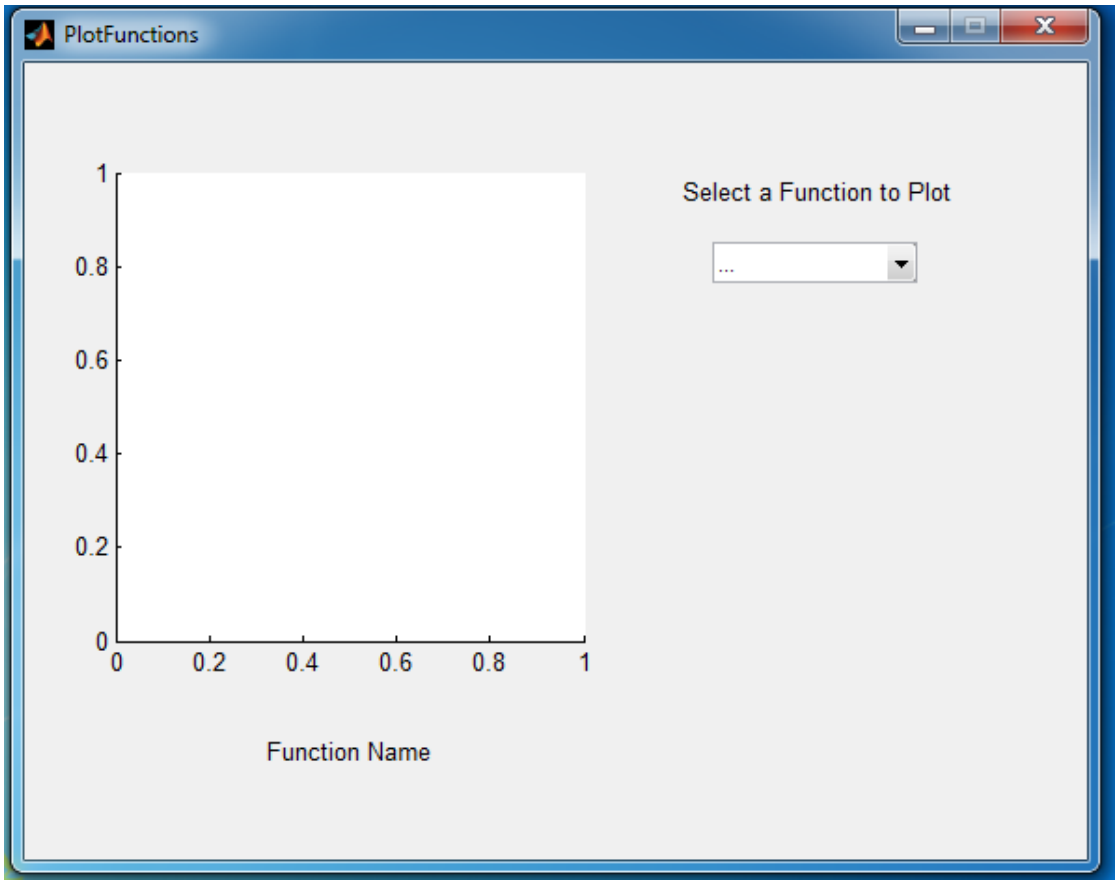


قم بوضع العناصر السابقة، ضمن static text الواقعة فوق القائمة المنسدلة قم بتغيير string واكتب الجملة Select a Function to Plot وابحث عن font size واجعله يساوي 10 عوضاً عن 8. كذلك الأمر بالنسبة لحجم الخط فقط بالنسبة ل static text الواقعة تحت axes وقم بتغيير tag الخاص بها إلى MessageToShow عوضاً عن text وغير الاسم string إلى Function Name.

ضمن Pop-up menu قم بتغيير string وضع فيه خيارات التوابع الممكن رسمها غضافةً للخيار الأول هو ... أي اننا لم نختر أي تابع كما في الشكل، وقم بتغيير tag إلى pop_plot



عند تشغيل الواجهة تجد الشكل التالي:

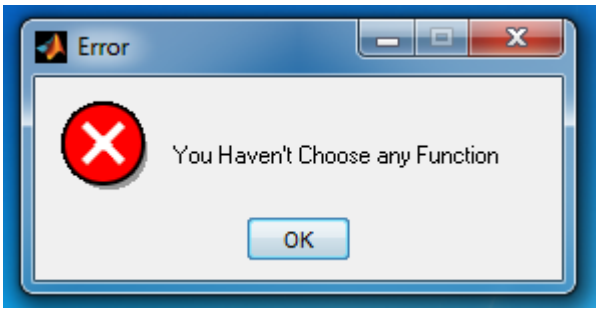


سننتقل الآن إلى برمجة التوابع أولاً التابع pop_plot_Callback

```
% --- Executes on selection change in pop_plot.
function pop_plot_Callback(hObject, eventdata, handles)
% hObject    handle to pop_plot (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
t=0:pi/100:2*pi;
state=get(handles.pop_plot,'value');
str = get(handles.pop_plot, 'String');
switch state
   case 1
      errordlg('You Haven''t Choose any Function','Error');
   case 2
      set(handles.MessageToShow,'string','You Plot Sin Function')
      y=sin(t);
   case 3
      set(handles.MessageToShow,'string','You Plot Cos Function')
      y=cos(t);
   case 4
      set(handles.MessageToShow,'string','You Plot Exp Function')
      y=exp(t);
end
axes(handles.axes1)
if state~=1
   plot(t,y);grid on;title(strcat(str{state},' Function'));
end
```

أولاً نقوم بتعريف المجال الزمني، ثم نعرف المتحول state سيأخذ القيمة المعادة لدى اختيار أحد القيم الموجودة في عنصر pop_up menu حيث سيتم تنفيذ إحدى الحالات التالية بحسب القيمة المرجعة:

1. عندها يتم إظهار رسالة خطأ على الشكل التالي تحتوي داخلها على العبارة الموضحة بالشكل.

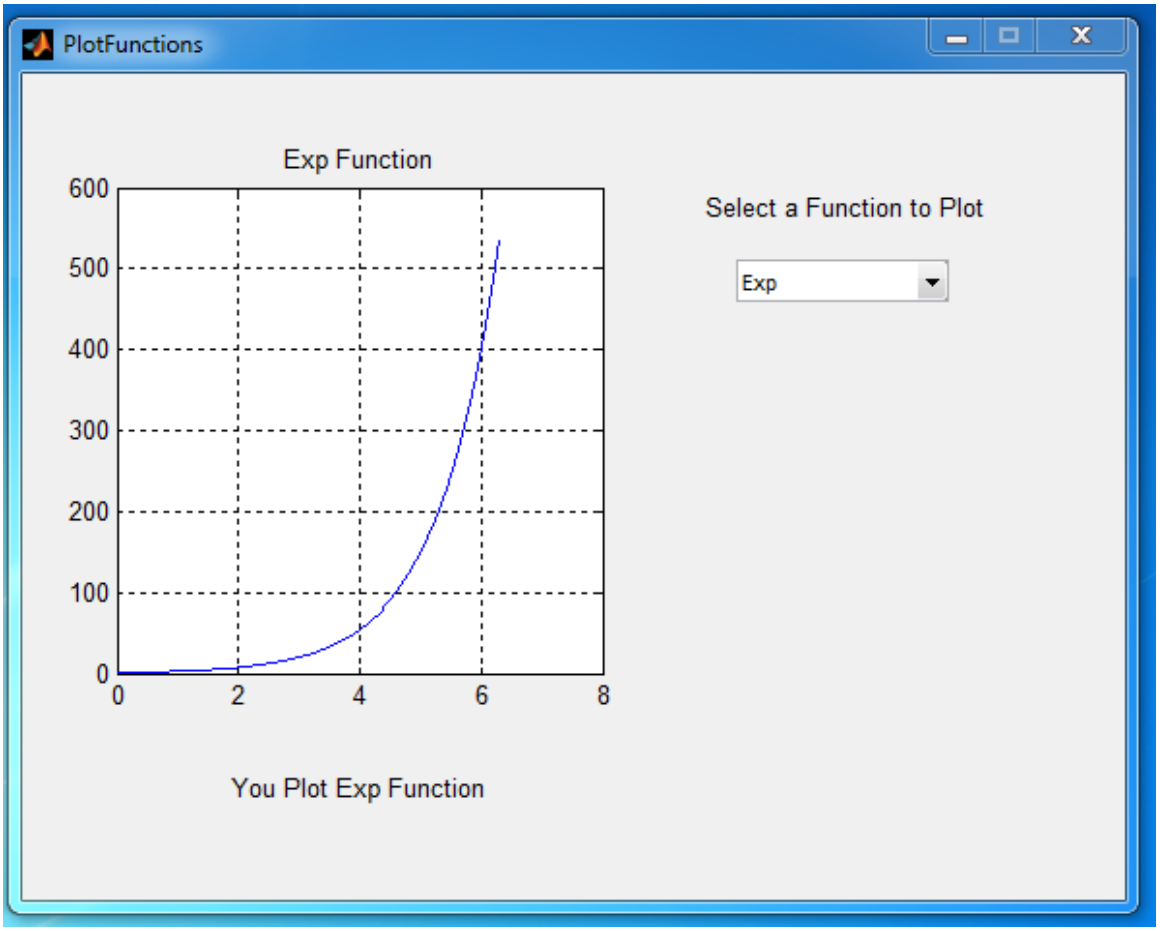


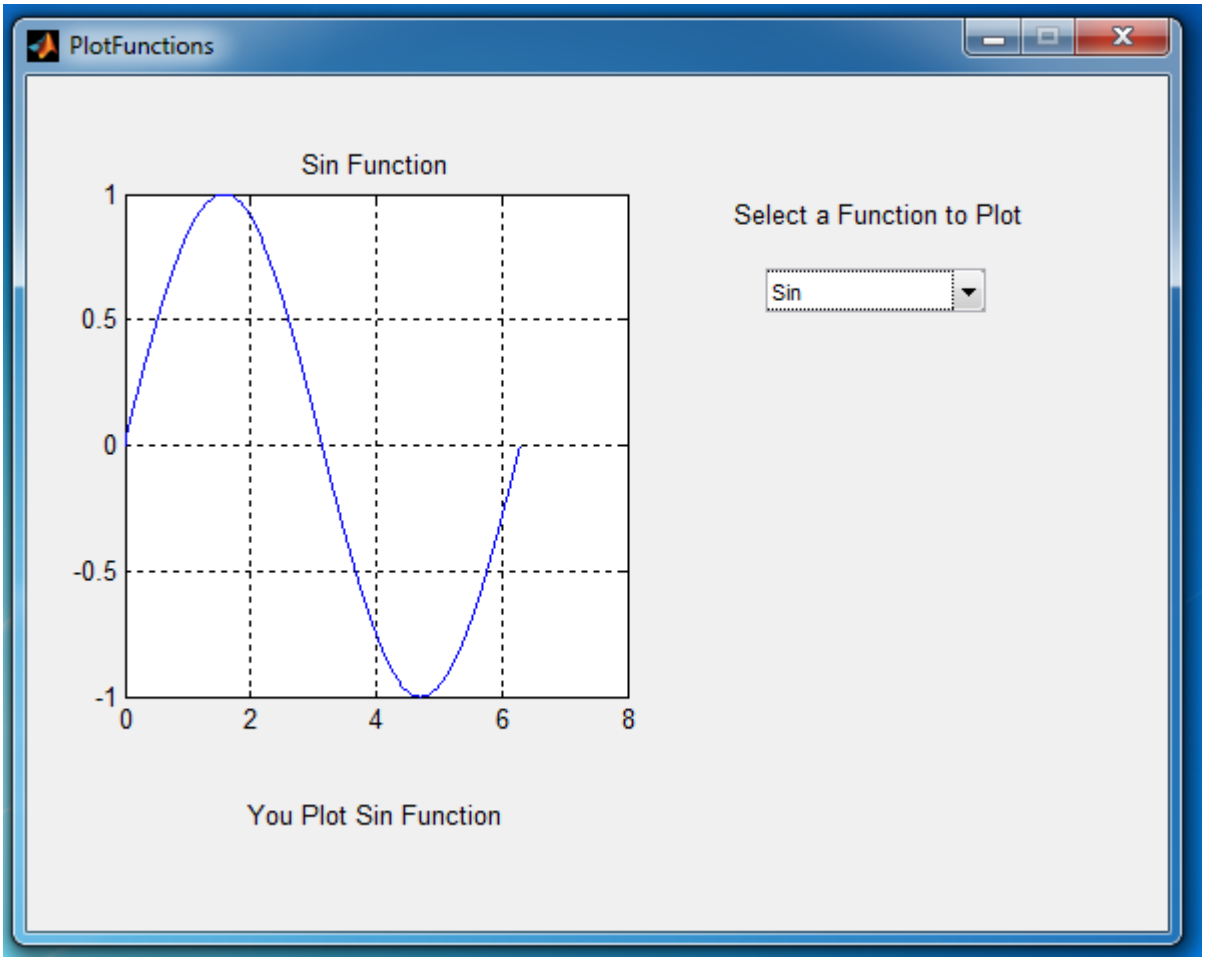
2. سيتم تعريف التابع sin(t) حيث الشعاع الزمني t معرف مسبقاً إضافةً لوضع سلسلة محرفية ضمن static text الموجودة أسفل محاور الرسم.
3. سيتم تعريف التابع cos(t) حيث الشعاع الزمني t معرف مسبقاً إضافةً لوضع سلسلة محرفية ضمن static text الموجودة أسفل محاور الرسم.
4. سيتم تعريف التابع exp(t) حيث الشعاع الزمني t معرف مسبقاً إضافةً لوضع سلسلة محرفية ضمن static text الموجودة أسفل محاور الرسم.

ثم سنقوم بتحديد العنصر axes1 لرسم منحني التابع المحدد عن طريق التابع axes

أخيرا نقوم بالرسم باستخدام التابع plot وإضافة عنوان في حالة كانت القيمة state لا تساوي الواحد، حيث انها توافق عدم اختيار تابع، وسنضع أيضاً عنوان للتابع نحصل عليه من خلال المتحول str والذي يعيد السلسلة المحرفية المختارة من القائمة المنسدلة ونستخدم تابع strcat من أجل دمج سلسلتين محرفيتين.

قم بالتنفيذ وتجريب الحالات المختلفة، مثال لأحد الحالات


PlotFunctions
Exp Function
600
500
400
300
200
100
0
0
2
4
6
8
Select a Function to Plot
Exp
You Plot Exp Function



لجعل البرنامج أعقد وأكثر فاعلية سنجعل المستخدم إما يحدد المجال الزمني باستخدام عنصر radio button ثم يدخل حدود المجال ضمن عنصرين من edit text أو لا يقوم بتحديد المجال فيؤخذ المجال الزمني المعرف مسبقاً.

يصبح شكل الواجهة كما يلي:

نحتاج إلى Panel لجعل مظهر الواجهة يبدو أكثر رتابةص ووضوحاً ولتنظيم الحقول ضمنها، إضافةً ل radio button و ثلاثة عناصر static text و ثلاثة عناصر edit text.

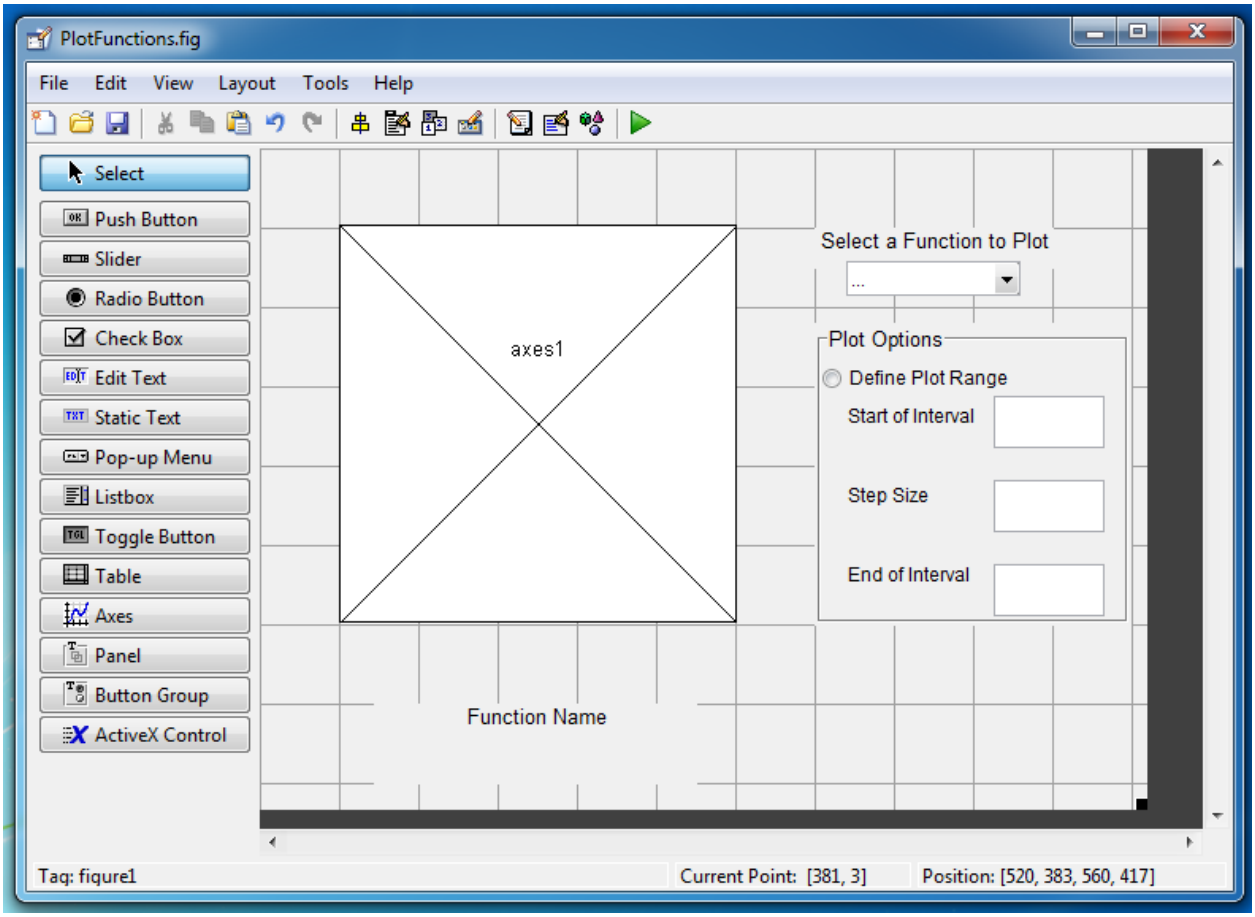
نغير اسم Panel كما في الشكل وحجم الخط إلى 10.

نغير اسم Radio Button إلى Define Plot Range ونغير tag له إلى ChooseOptions وحجم الخط إلى 9.

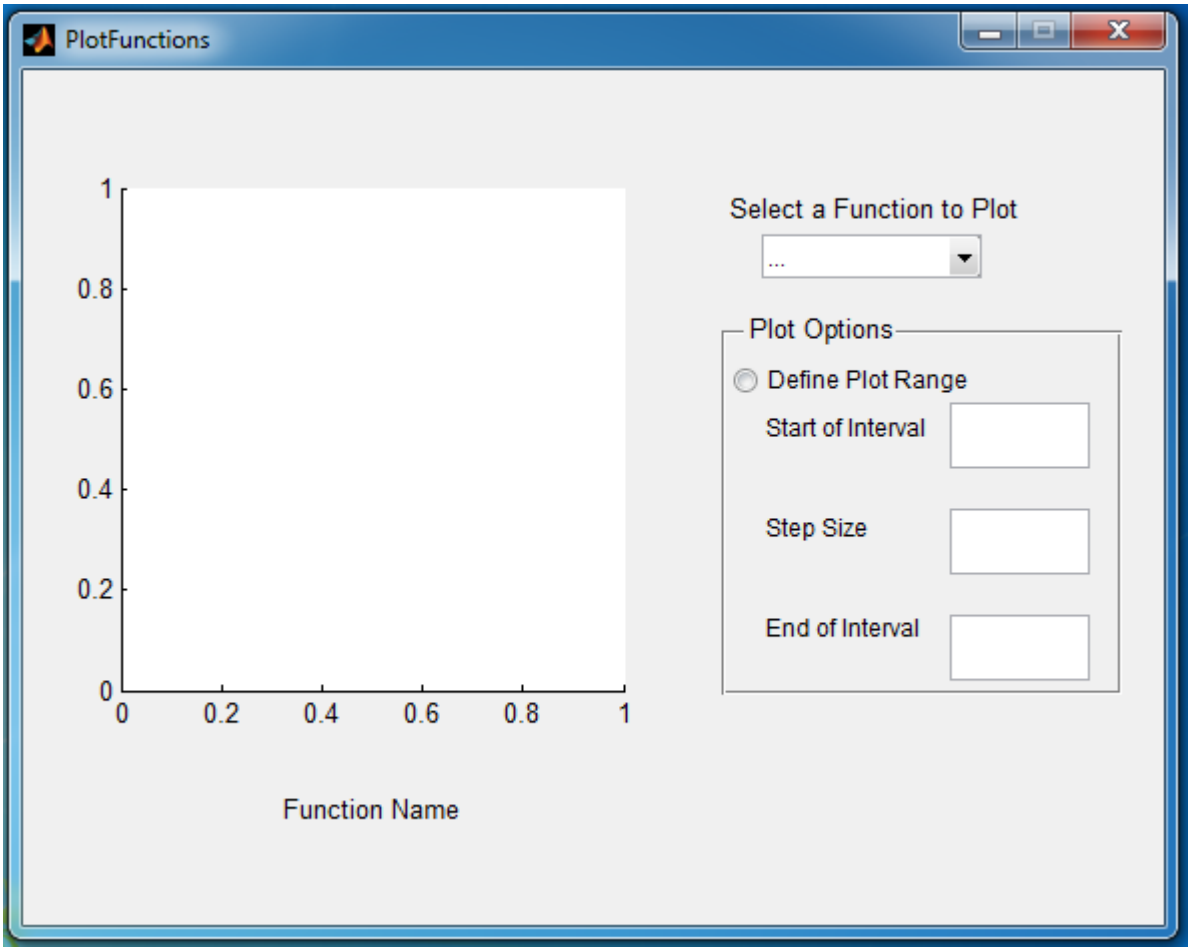
نغير أسماء static text كما هو موضح في الشكل للتعبير عن بداية المجال والخطوة والنهاية المجال إضافةً لتغيير حجم الخط إلى 9 ويرجى محاذاة العناصر للحصول على شكل أنيق. كذلك نضع لهم Tag هو على الترتيب start, step, stop.

نغير اسم كل من edit text إلى null أي لا نضع أي قيمة. ونغير tag إلى Start_Int لأول عنصر

نعيد الأمر للثانية الموافقة للخطوة ونغير tag إلى Step_Int أما الأخيرة فنغيرها إلى End_Int ويرجى محاذاة العناصر للحصول على شكل أنيق.



عند التشغيل نحصل على الشكل:



نرغب أولاً في عدم إظهار خيارات إدخال المجال إلا في حال قام المستخدم باختيار الخيار Define Plot Range.

أولاً من أجل عدم إظهار الجمل التوضيحية وصناديق إدخال القيم يمكن كتابة الرماز التالي ضمن التابع PlotFunctions_OpeningFcn

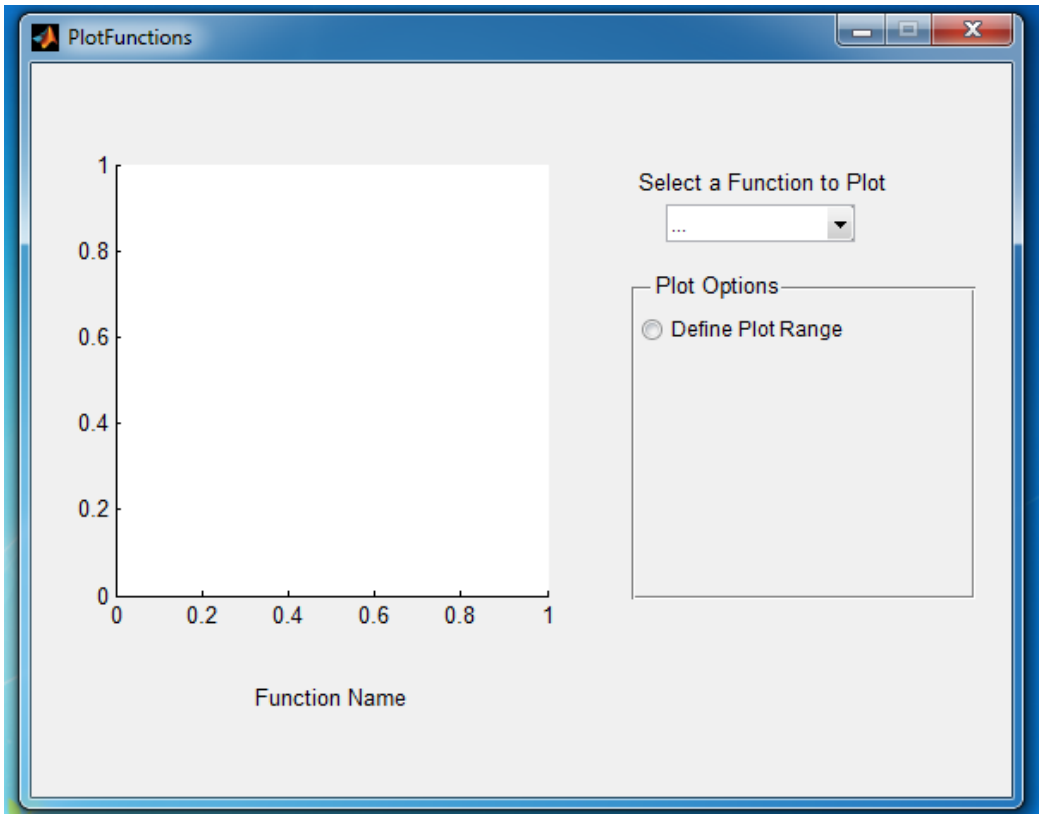
يصبح التابع:

```
% --- Executes just before PlotFunctions is made visible.
function PlotFunctions_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to PlotFunctions (see VARARGIN)
set(handles.start,'Visible','off');
set(handles.step,'Visible','off');
```

```
set(handles.stop,'Visible','off');
set(handles.Start_Int,'Visible','off');
set(handles.Step_Int,'Visible','off');
set(handles.End_Int,'Visible','off');
% Choose default command line output for PlotFunctions
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);
```

أضفنا فقط 6 سطور برمجية تجعل العناصر مختفية من خلال خيار 'Visible','off'.

عند التشغيل يصبح الشكل:



يجب الآن عند الضغط على خيار Define Plot Range أن تظهر العناصر السابقة وعند عدم اختياره يجب اختفاء العناصر، لذلك نضيف الرماز التالي:

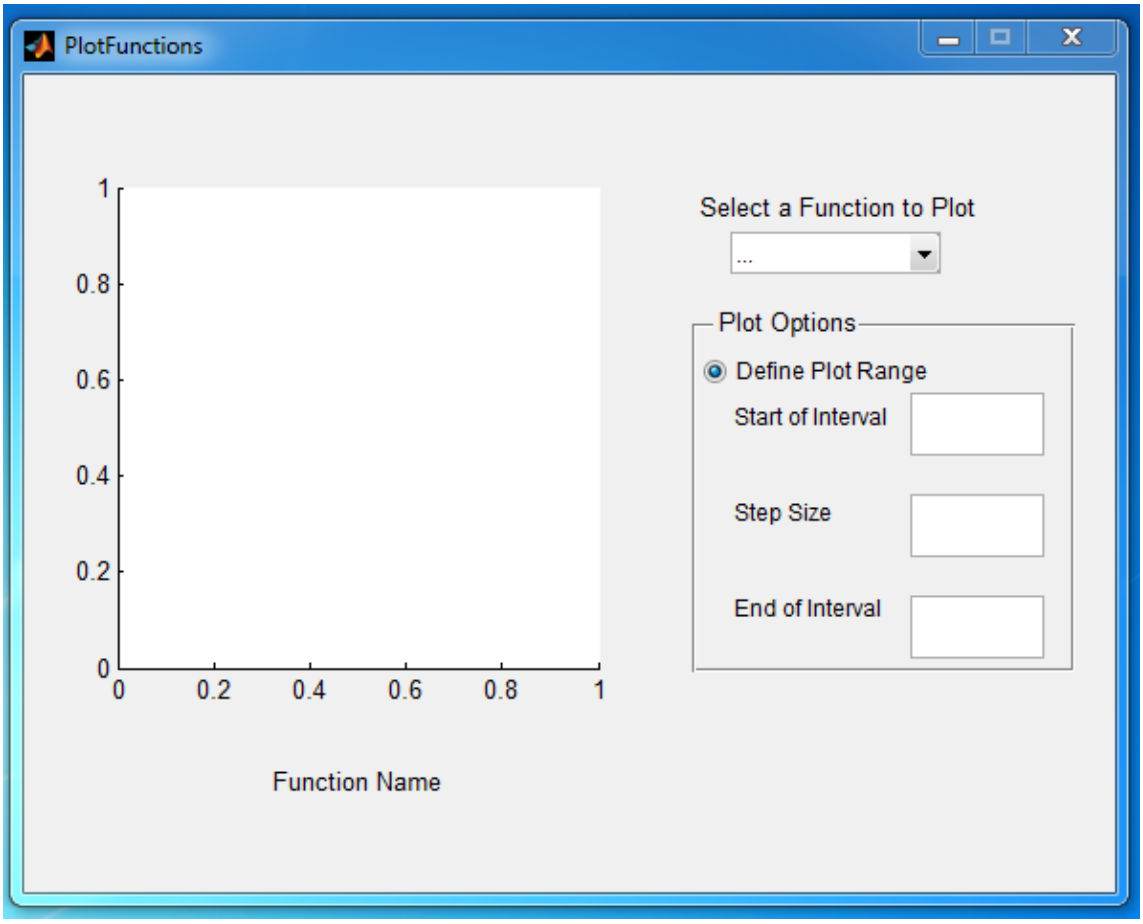
```
% --- Executes on button press in ChooseOptions.
function ChooseOptions_Callback(hObject, eventdata, handles)
% hObject    handle to ChooseOptions (see GCBO)
```

```
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
s=get(handles.ChooseOptions,'value');
if s==0
   set(handles.start,'Visible','off');
   set(handles.step,'Visible','off');
   set(handles.stop,'Visible','off');
   set(handles.Start_Int,'Visible','off');
   set(handles.Step_Int,'Visible','off');
   set(handles.End_Int,'Visible','off');
else
   set(handles.start,'Visible','on');
   set(handles.step,'Visible','on');
   set(handles.stop,'Visible','on');
   set(handles.Start_Int,'Visible','on');
   set(handles.Step_Int,'Visible','on');
   set(handles.End_Int,'Visible','on');
end
```

حيث نختبر القيمة المعادة من عنصر radiobutton في المتحول s في حال كانت:

- 0 فهذا يعني أن الزر غير مضغوط وبالتلي سنحافظ على بقاء العناصر الأخرى مختفية
- 1 فهذا يعني أن الزر مضغوط وبالتالي يجب إظهار العناصر لإدخال القيم العدّدية الموصفة للمجال.

عند التنفيذ والضغط على الزر نجد الشكل التالي:



سنقوم بمناقشة بعض الحالات من أجل دخل المربعات التي تمثل بداية ونهاية المجال والخطوة.

ضمن التابع التالي

```
% --- Executes on selection change in pop_plot.
function pop_plot_Callback(hObject, eventdata, handles)
% hObject    handle to pop_plot (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
t=0:pi/100:2*pi;
state=get(handles.pop_plot,'value');
str = get(handles.pop_plot, 'String');
```

بعد السطر السابق أضف الرماز التالي:

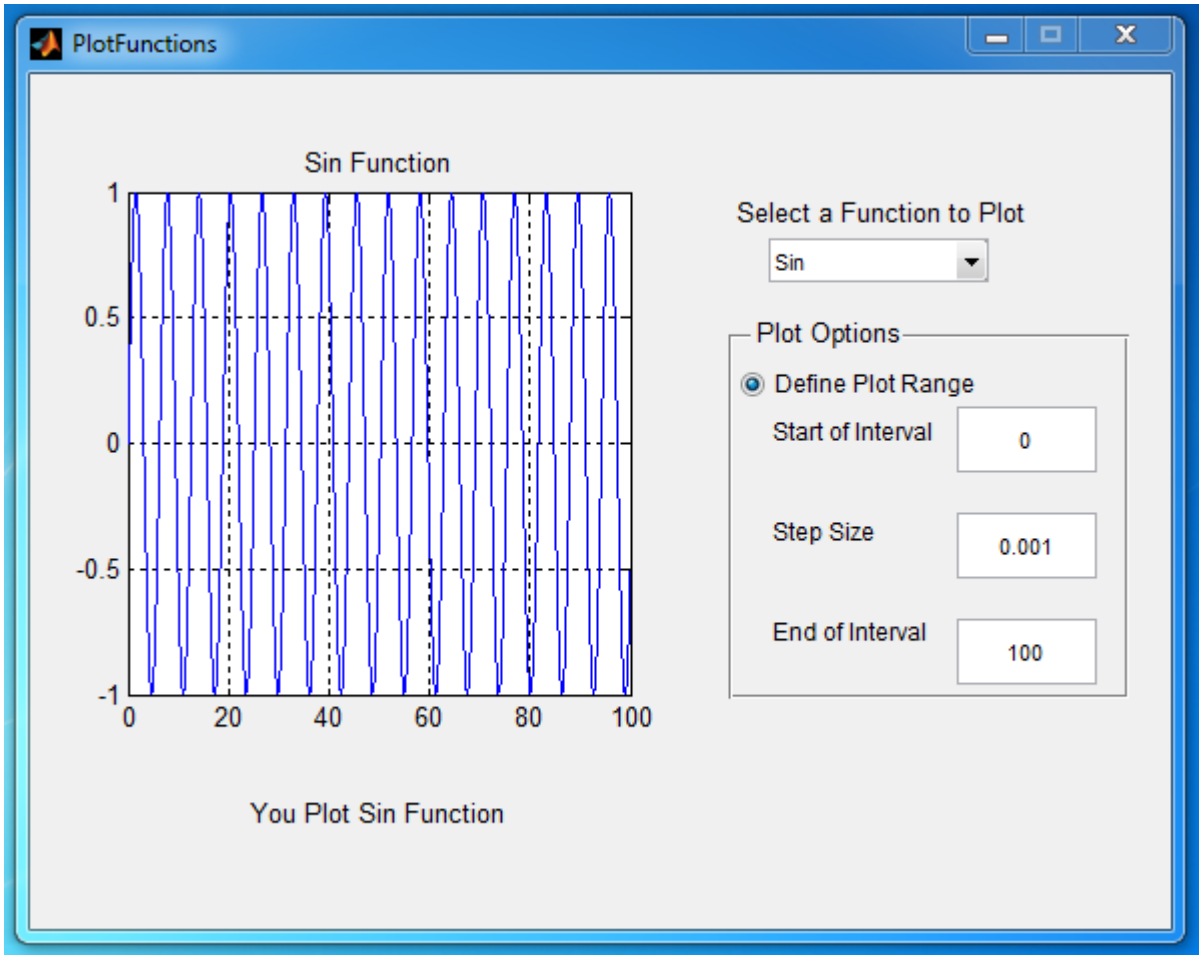
```
s=get(handles.ChooseOptions,'value');
if s==1
   startInt=str2double(get(handles.Start_Int,'string'));
```

```
    step=str2double(get(handles.Step_Int,'string'));
    endInt=str2double(get(handles.End_Int,'string'));
    if isnan(startInt)==1 || isnan(step)==1 || isnan(endInt)==1 ...
            || isinf(startInt)==1 ||  isinf(step)==1 || isinf(endInt)==1
        errordlg('Invalid Arguments');
    elseif step <=0  || step> (endInt-startInt)
        errordlg('Invalid Step');
    else
        t=startInt:step:endInt;
    end
end
```

حيث نختبر القيمة المعادة من عنصر radiobutton في المتحول s في حال كانت مساوية للواحد أي أن الزر مضغوط فهذا يعني أن المستخدم قد اختار تحديد حدود المجال الزمني حيث نختبرهما في البداية إذا كانت القيم المدخلة داخل عناصر edit text فارغة أو قيمة أي منها لانهائية أو في حال كون الخطوة أكبر من المجال أو أن الخطوة سالبة فعندها سنظهر رسالة خطأ وإلا سيكون المجال الزمني تبعا لما قام المستخدم بإدخاله.

يظهر التنفيذ كما هو موضح في الشكل التالي في إحدى الحالات:

قم بتجريب الحالات المختلفة



يمكن إضافة العديد من التفاصيل ومناقشة عدد كبير من الحالات لجعل برنامج السابق أكثر فاعلية.