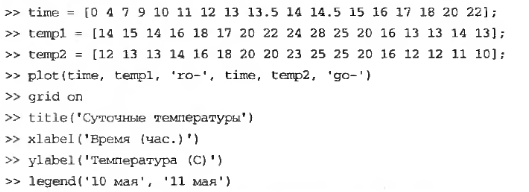
**Лекція 18. ПРОГРАМНІ ЗАСОБИ ВІДОБРАЖЕННЯ ВИМІРЮВАЛЬНОЇ ІНФОРМАЦІЇ. ВИКОРИСТАННЯ ГРАФІЧНИХ МОЖЛИВОСТЕЙ ПРОГРАМИ MATLAB**

1. **Оформлення графіків**

На минулому занятті ми навчилися будувати графіки за допомогою функції plot. Однак слід зазначити, що при побудові графіків достатньо вагомим є оформлення та представленні інформації, як наприклад, координатна сітка, підписи осей, заголовки елементів, наявність легенди. Такі можливості реалізуються або за допомогою додаткових параметрів або за допомогою команд та функцій. Перечислимо основні із них. Сітка наноситься командою **grid on**, функції **xlabel** **ylabel** служатьдля розміщення підписів осей, а **title** – для заголовку. Для відображення легенди використовують функцію **legend**.

Наприклад, виведемо результати виміру добової температури:



В результаті отримаємо наступний вид графіку, рис.1

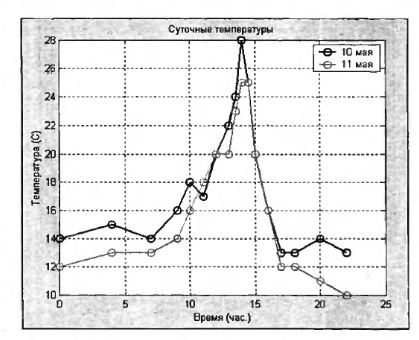


Рис.1

1. Команда hold on дозволяє додавати криві на уже існуючий графік. Функції subplot дозволяє виводити декілька графіків в одному вікні. Так, наприклад, при введенні наступних команд, що представлено на рис.2, а результат на рис.3

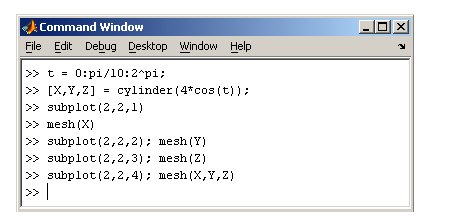


Рис.2

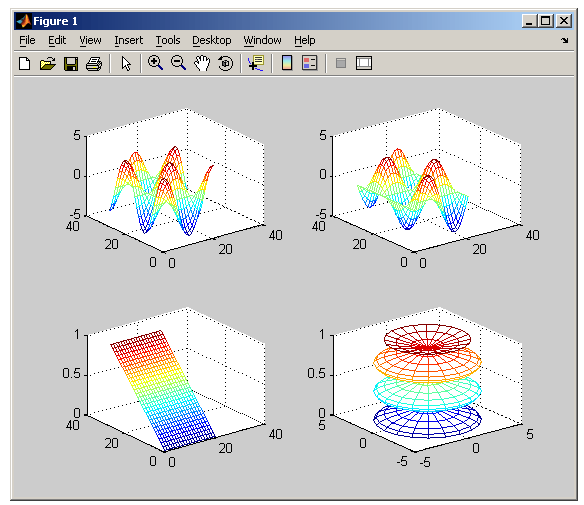


Рис.3

1. Графіки параметричних на кусково-заданих функцій

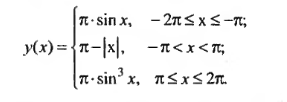
Для побудови функцій, що задані параметрично, необхідно спочатку згенерувати вектор значень аргументу, а потім визначити значення функцій та записати їх у вектори, які слід використовувати як аргументи функції plot.

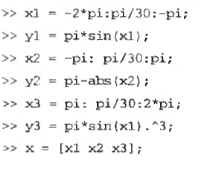
Так, наприклад, для побудови графіку функції x(t)=0.5sint, y(t)=0.7cost для необхідно задати наступні команди:

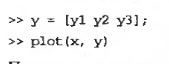




Для побудови кусково-заданої функції необхідно вирахувати кожну вітку, а потім об’єднати значення абсцис в один вектор, а значення ординат в інший вектор, а потім вивести графік функцій цих двох масивів. Наприклад,







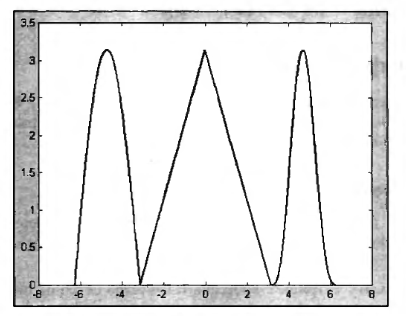
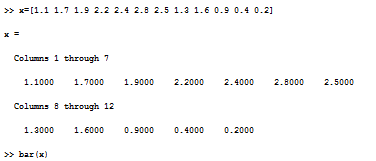


Рис.4

1. **Представлення векторних даних**

Відображення векторних даних у вигляді стовпчикової діаграми відбувається за допомогою функції **bar**. Наприклад, маємо вектор:





Як видно з рисунку на екрані з’являється стовпчикова діаграма. По горизонтальній осі відкладається номер елемента, а по вертикалі його значення. В якості аргументу може використовуватися як вектор-рядок так і вектор-стовпець.

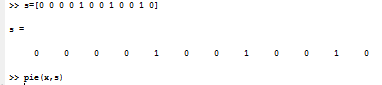
Функція **barh** будує горизонтальну стовпчикову діаграму, що повернута на 900, а **bar3** використовується для побудови об’ємних діаграм. Використовуються аналогічно до функції bar.

Побудова кругової діаграми відбувається із залученням функції **pie.** Наприклад: >>pie(x). Результат представляє собою таку кругову діаграму:**огічно до побудови обємних повернута на 90вертикалі його значення.х знаходяться и рівною кількості рядків другої мат**



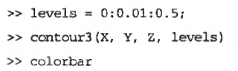
В даному випадку площа секторів відповідає процентному вкладу кожного із елементів вектора в загальну суму, тобто проводиться нормування.

При необхідності виокремлення певного сектору діаграми необхідно задати другим аргументом функції **pie** вектор, що складається із одиниць та нулів, при чому одиниця стоїть в позиції, що відповідає виокремленій частині. Для візуалізації використовується також функція **pie3**

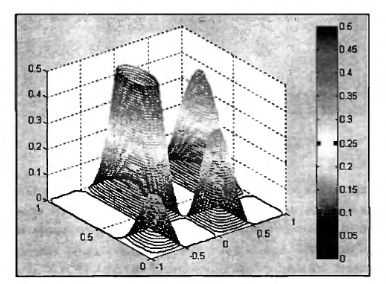
 

1. **Контурні графіки**

Matlab дозволяє будувати поверхні, що складаються із ліній рівня за допомогою функції contour3. Число ліній рівня задається автоматично. Є можливим задати четвертим аргументом або число ліній рівня або вектор, елементи якого рівні значенням функції, що будуть відображатися у вигляді ліній рівня. Задавання вектора доцільне, коли необхідно дослідити поведінку функції в деякій області її значень (зріз функції). Наприклад, побудуємо поверхню, що складається з ліній рівня, що відповідають значенням функції від 0 до 0,5 з кроком 0,01

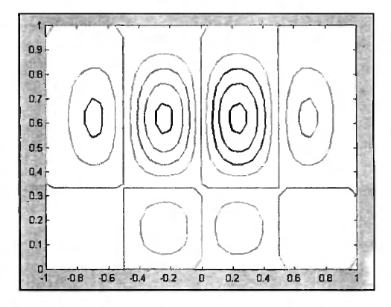


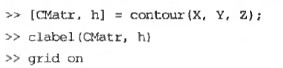
В результаті отримуємо:

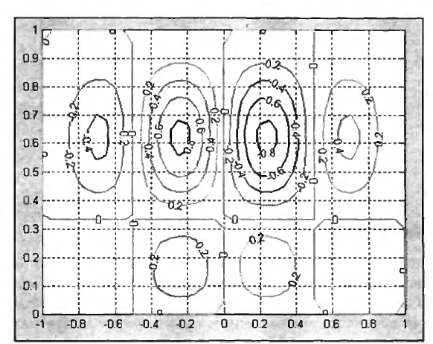


Для отримання контурних графіків використовується функція **contour** та **contour**

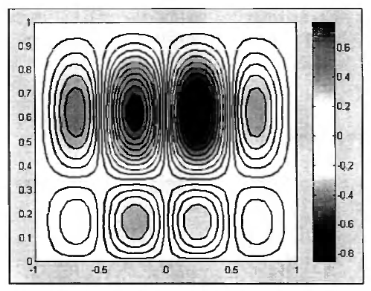
Наприклад,









1. **Додавання формул на графіки**

Досить часто необхідним є розміщення формули в заголовку графіку або поруч із вертикальною віссю. Для розміщення підпису біля вертикальної осі використовується команда **zlabel**.

Використання в аргументах команд деяких математичних позначень в форматі ТеХ дозволяє додавати формули на графік.

Якщо формула не поміщається в один рядок при наборі команди, то слід використати оператор три крапки. Наприклад, для відображення формули в заголовку функції:



слід набрати наступну команду:



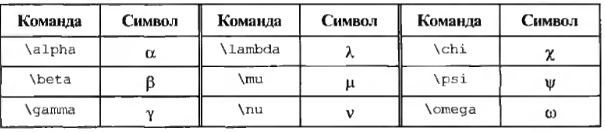
Правила набору формул та зміна властивостей шрифтів наведені в таблиці 10.1

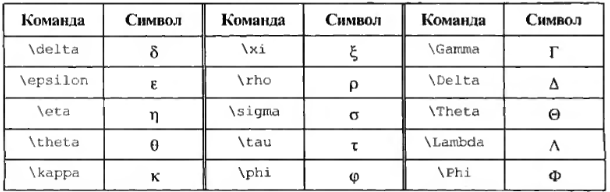
Таблиця 10.1 – Правила набору формул та зміна властивостей шрифтів

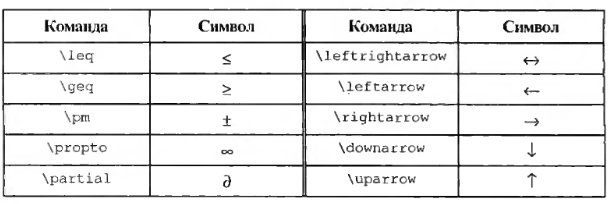
|  |  |  |
| --- | --- | --- |
| Що вимагається? | Команда ТеХ | Результат |
| Виділення ***курсивом*** одного символу або тексту | {\itx} | *x* |
| 1.2{\itP} | 1.2*P* |
| {\itГіперболічний} синус | *Гіперболічний* синус |
| Виділення жирним шрифтом одного символу або тексту | Шаблон матриці {\bfM} | Шаблон матриці **М** |
| {\bfАЧХ} фільтра | **АЧХ** фільтра |
| Набір символу або тексту жирним курсивом | Вектори {\bf\itx} та {\bf\ity} | Вектори ***x*** та ***y*** |
| {\bf\itОптимальна} крива | ***Оптимальна*** крива |
| Зміна шрифта та його розміру | {\fontname{arial}\fontsize{14}z-функція} | Z-функція |
| Степінь, верхній індекс | x^{2} | x2 |
| {\itx}^{2.5} | *x*2.5 |
| {\ite}^{\it-x} | *e-x* |
| Нижній індекс | f\_{5} | f5 |
|  | f\_{\itxx} | f*xx* |

Можливе використання грецьких літер та спеціальних символів, наприклад title(‘Залежність при a=\pi’) призводить до заголовку «Залежність при а=π». В таблиці 10.2 приведено команди ТеХ для вставки деяких прописних і рядкових грецьких літер і спеціальних символів.

Таблиця 10.2 – Грецькі літери та спеціальні символи



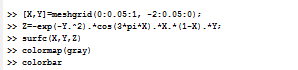




Так, розглянемо приклад побудови функції:



Введемо наступні команди в у вікні Command Window:





В результаті отримуємо графік функції з вказаними підписами осей та заголовком:

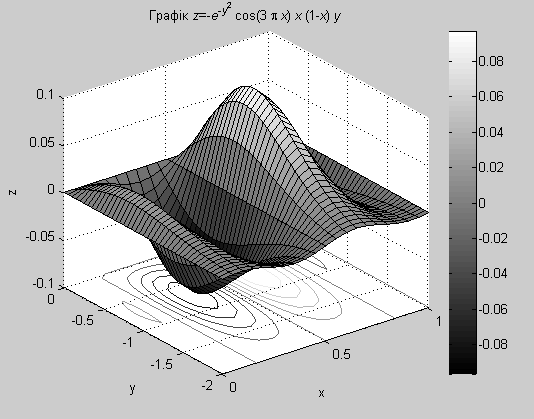
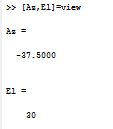


Рисунок 10.1

1. **Анімаційні графіки**

Для отримання повної інформації про поверхню та огляду її зі всіх сторін необхідно можна змінювати положення спостерігача. Положення спостерігача характеризується двома кутами: азимутом (Az) та кутом піднесення (El). Азимут вираховується від осі, що протилежна ***y***, а кут піднесення від площини ***xy***. Зміна положення спостерігача відбувається за допомогою функції **view**. Аргументами **view** є азимут та кут піднесення, що обраховується в градусах. По замовчуванню:

Az=–37,50, El=300



Задаємо параметри Az=1350, El=450



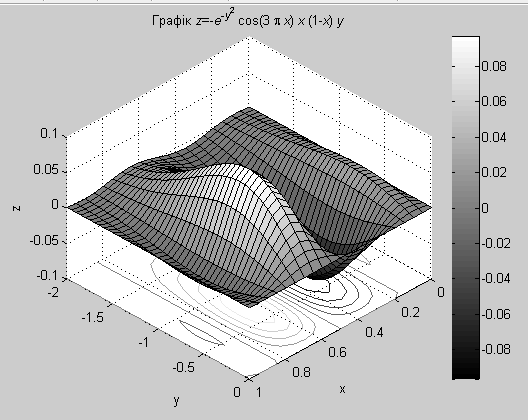


Рисунок 10.2 – Графік з точки зору спостерігача Az=1350, El=450

При *view(0,0)*

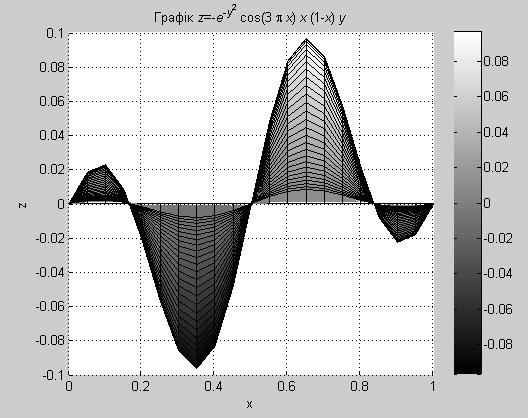


Рисунок 10.3 **–** Графік з точки зору спостерігача Az=00, El=00

Для побудови анімаційних графіків використовується функції comet та comet3. Так наприклад, побудуємо траєкторію руху точки протягом 10 секунд, координати яких змінюються по закону:



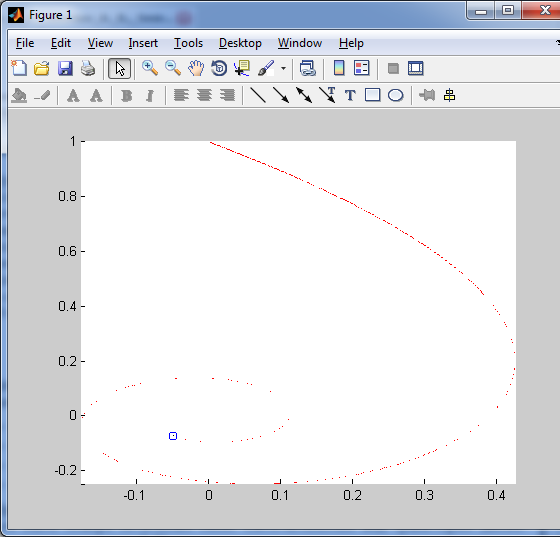
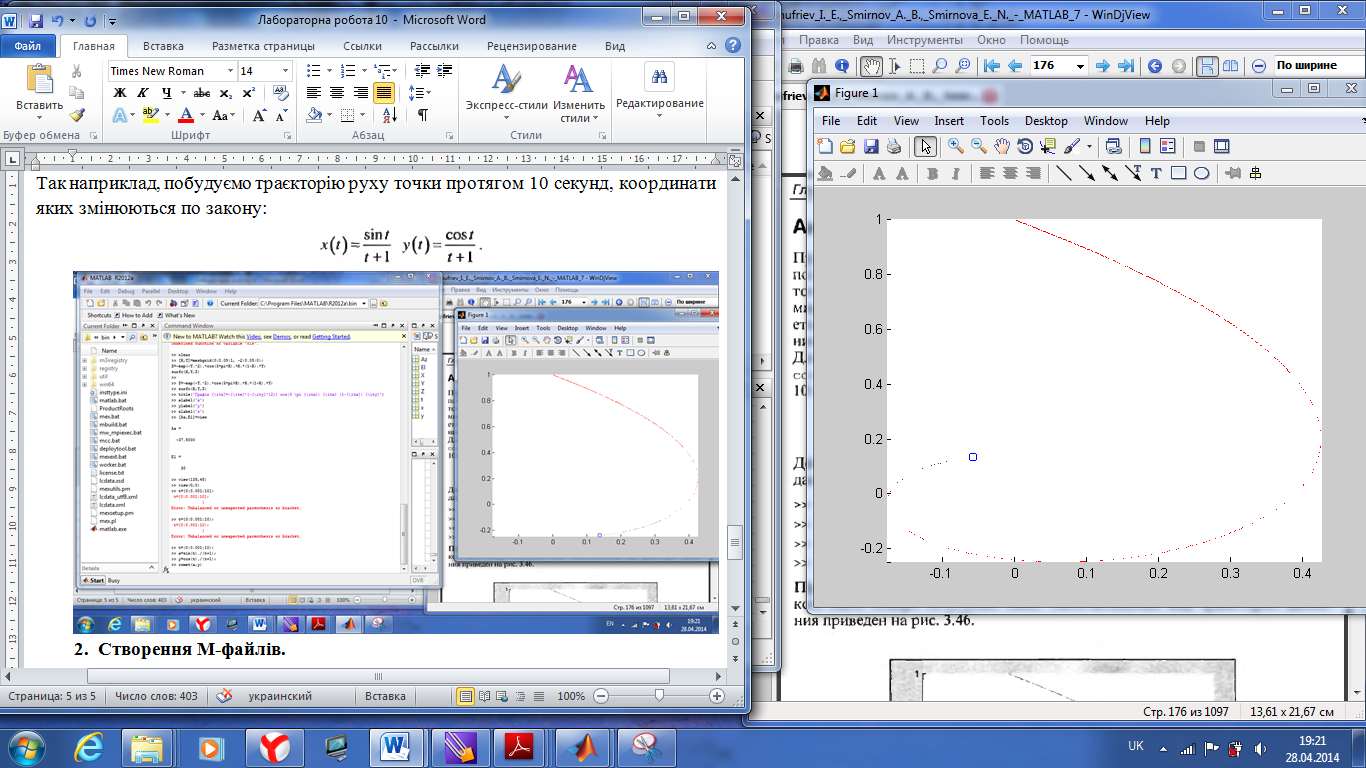
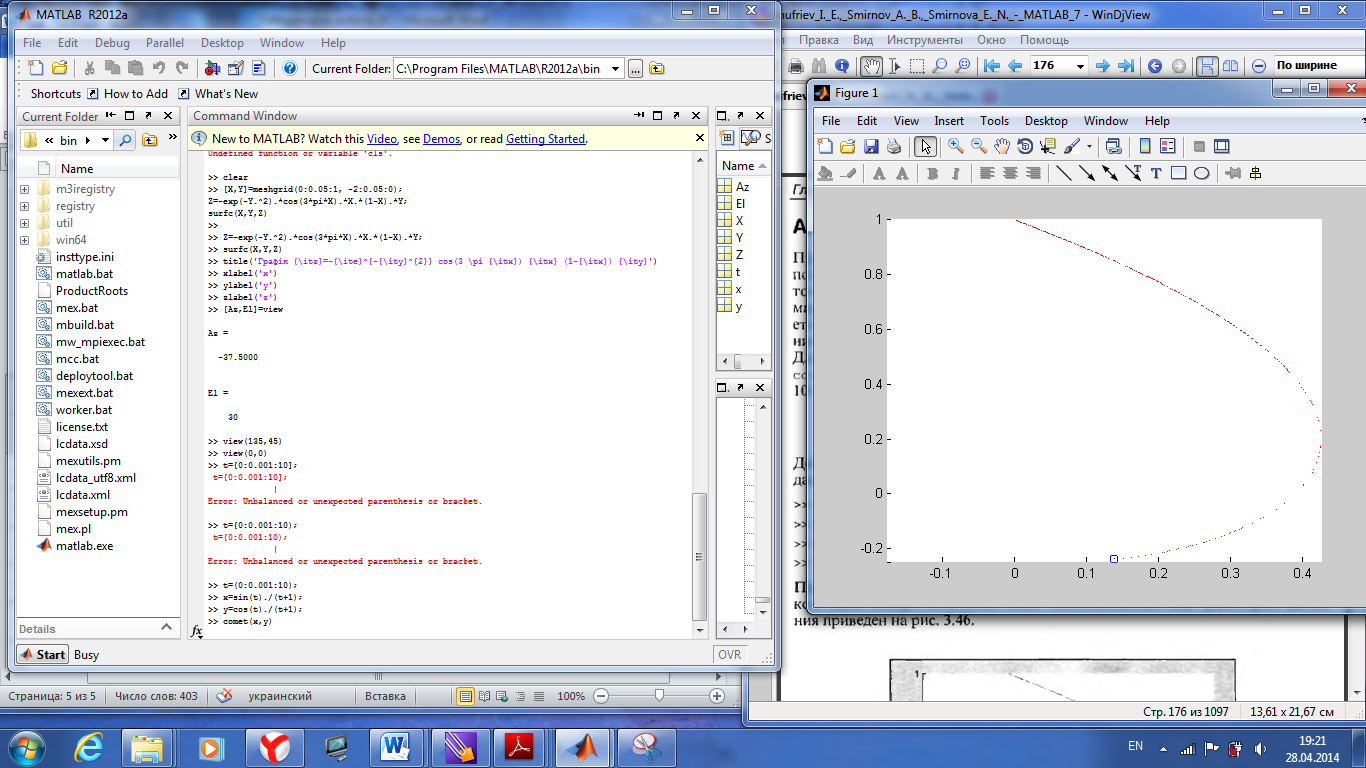


Рисунок 10.4 – Вигляд траєкторії руху точки

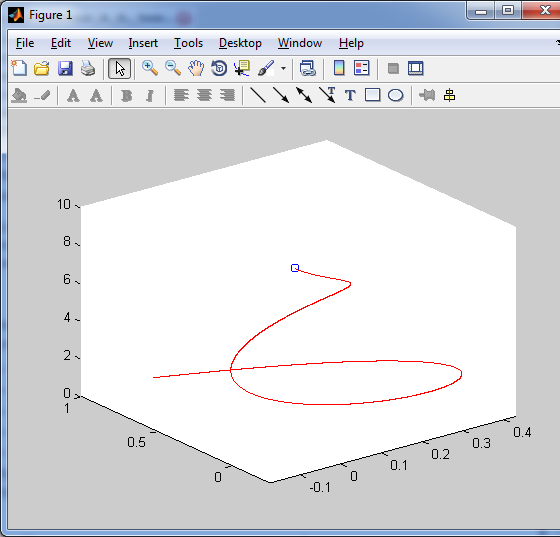


Рисунок 10.5 – Кінцевий вигляд траєкторії руху (comet3)

1. **Візуалізація векторних полів**

Функції **compass**, **feather** та **quiver** графічно представляють сукупність двомірних векторів. В якості прикладу побудуємо залежність вектора швидкості тіла, що кинуте під кутом до горизонталі, від часу:

, ,

де , – проекції вектора початкової швидкості; m – маса; g – прискорення вільного падіння; t – час. Вектори повинні виходити із точок, що належать траєкторії руху тіла. Траєкторія руху описується законом:

Приймаємо та тривалість польоту складає дві секунди до падіння тіла на поверхню і прослідкуємо за зміною його швидкості через кожні 0,2 секунди.







Вигляд графіку представлено на рис. 10.6.

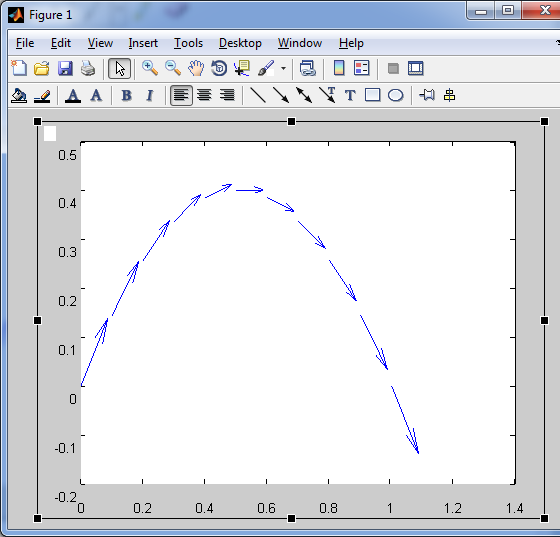


Рисунок 10.6

Функції compass та feather реалізує дещо інші способи графічного відображення векторних полів.

>> compass(ux, uy)

>> feather(ux, uy)