ЛЕКЦИЯ № 4 · Проект MATLAB IoT

Средства MATLAB для построения двумерных и трёхмерных графиков: plot, plot3, mesh и surf



Бельгибаев Бауржан Абдрахимович
Д.т.н., доцент
Кафедра «Искусственный интеллект и big data»
ОП 6B07113 «Интеллектуальные системы управления»

Цель лекции

Сформировать знания и практические навыки визуализации числовых, экспериментальных и модельных данных средствами MATLAB Online; освоить построение двумерных графиков и трёхмерных линий и поверхностей, их оформление, интерпретацию и применение к задачам IoT и интеллектуального управления.

План лекции

1. Роль визуализации в анализе инженерных и IoT-данных
2. Организация графического окна MATLAB и функция plot
3. Оформление осей, подписей, легенд и нескольких кривых
4. Дискретные измерения, scatter, stem и обработка пропусков
5. Построение трёхмерных траекторий функцией plot3
6. Создание координатных сеток с помощью meshgrid
7. Каркасные поверхности функцией mesh
8. Цветные поверхности функцией surf
9. Выбор масштаба, цветовой карты и ракурса
10. Примеры визуализации температуры, траектории робота и карты параметров
11. Требования к научным графикам и типичные ошибки

1 Роль визуализации в инженерных исследованиях

Визуализация данных — это представление числовой информации в графической форме для выявления закономерностей, отклонений и взаимосвязей. В системах управления график позволяет наблюдать переходный процесс, сравнивать сигнал с заданием и анализировать качество регулирования. В IoT графики служат основой удалённого мониторинга, диагностики и отчётности.

Различают экспериментальные данные, полученные с физического датчика, и расчётные данные, полученные из математической модели. Эти два источника нельзя смешивать без пояснений. Экспериментальные точки отражают дискретность и погрешности измерения, тогда как модельная кривая описывает принятые математические предположения.

График должен отвечать на конкретный вопрос. Для изменения температуры во времени подходит двумерная линия; для координат движения робота в пространстве — трёхмерная траектория; для зависимости температуры от двух независимых параметров — поверхность. Выбор неподходящего типа графика может затруднить интерпретацию.

В MATLAB визуализация опирается на массивы. Обычно вектор x содержит независимую переменную, а вектор y — значения функции или измерения. Для трёхмерной линии нужны три согласованных массива x, y, z, а для поверхности — координатные сетки X, Y и матрица Z.

2 Двумерные графики: функция plot

Функция plot(x,y) строит линию на плоскости по парным значениям двух векторов. Массивы x и y должны иметь совместимые размеры. Если задать только plot(y), MATLAB использует номера элементов как координаты по горизонтальной оси.

Например, модель температуры $T(t)=T_a+(T_0-T_a)\exp(-t/\tau)$ описывает экспоненциальное приближение температуры объекта к температуре среды. Здесь T_a — температура окружающей среды, T_0 — начальная температура, τ — постоянная времени, t — время. Аргумент экспоненты t/τ безразмерен.

Для вычисления значений на временной сетке удобно использовать оператор двоеточия или linspace. Чем меньше шаг, тем визуально плавнее линия. Однако увеличение количества расчётных точек не делает физическую модель точнее само по себе.

Команда figure создаёт графическое окно, grid on добавляет координатную сетку, xlabel и ylabel подписывают оси, title задаёт заголовок. Параметр LineWidth задаёт толщину линии, а строка вида «-o» добавляет маркеры исходных точек.

2 Двумерные графики: функция plot (продолжение)

```
t = 0:10:300;            % время, с
Ta = 22; T0 = 75; tau = 100;
T = Ta + (T0-Ta).*exp(-t./tau);
figure;
plot(t,T,'-','LineWidth',1.8);
grid on;
xlabel('Время, с'); ylabel('Температура, °C');
title('Модель охлаждения объекта');
```

При t=0 модель даёт 75 °C; при t=100 с температура равна 22+53/e≈41,50 °C. Линия иллюстрирует модель, а не результаты реального датчика.

3 Оформление графиков и сравнение нескольких зависимостей

Для сравнения нескольких процессов на одной диаграмме применяется hold on. После первой команды plot новые линии добавляются в то же окно. Команда hold off возвращает обычный режим. Функция legend связывает стиль линии с её физическим смыслом.

Оси следует подписывать с указанием единиц измерения. Например, «Время, с» и «Температура, °C». Если несколько линий имеют одинаковую физическую размерность, их можно разместить на общей оси. Смешивать температуру и давление на одной оси без специального обоснования нельзя.

Команды xlim и ylim ограничивают диапазоны осей. При выборе пределов важно не скрывать выбросы или существенные участки процесса. Для научных публикаций предпочтительны нейтральные читаемые цвета, различимые маркеры и понятные подписи.

Функция subplot или tiledlayout позволяет размещать несколько отдельных графиков в одном окне. Это полезно, если необходимо сопоставить температуру, влажность и давление, не смешивая разные единицы измерения.

Экспорт выполняется, например, командой exportgraphics(gcf,'temperature.png','Resolution',300). При подготовке отчёта важно сохранить исходный скрипт, чтобы график можно было воспроизвести.

3 Оформление графиков и сравнение нескольких зависимостей (продолжение)

```
t = 0:5:200;
Ta = 20; T0 = 70;
T1 = Ta+(T0-Ta)*exp(-t/60);
T2 = Ta+(T0-Ta)*exp(-t/120);
figure;
plot(t,T1,'-','LineWidth',1.8); hold on;
plot(t,T2,'--','LineWidth',1.8);
grid on; xlabel('Время, с'); ylabel('Температура, °C');
legend('tau = 60 c','tau = 120 c','Location','northeast');
title('Сравнение постоянных времени');
exportgraphics(gcf,'cooling_comparison.png','Resolution',300);
```

4 Дискретные измерения и визуализация точек

Датчик выдаёт конечное количество отсчётов, поэтому важно показывать реальные точки измерения. Функция plot(t,T,'-o') соединяет точки линией, scatter(t,T) строит отдельные маркеры, а stem(t,T) отображает дискретные значения в виде вертикальных линий.

Соединение измерений отрезками не означает, что между соседними отсчётами физическая величина изменялась строго линейно. Это лишь способ визуального представления. При больших интервалах опроса промежуточное поведение может оставаться неизвестным.

Если часть данных отсутствует, в массиве могут появляться NaN. При построении графика MATLAB обычно разрывает линию в местах таких значений. Не следует автоматически заменять все пропуски нулями: это создаёт ложные провалы.

Для выявления аномалий можно добавить горизонтальную пороговую линию с помощью yline. Но порог должен соответствовать технологическим требованиям, а интерпретация измерения — учитывать погрешность датчика и задержку передачи.

Пример: значения температуры [25, 25.4, 25.8, NaN, 27.0, 28.2] °C получены с интервалом 10 с. Пропуск на 30-й секунде следует отметить в журнале и не интерпретировать как температуру 0 °C.

4 Дискретные измерения и визуализация точек (продолжение)

```
t = 0:10:50;
T = [25 25.4 25.8 NaN 27.0 28.2];
figure;
plot(t,T,'-o','LineWidth',1.5,'MarkerSize',6);
yline(28,'--','Порог 28 °C');
grid on; xlabel('Время, с'); ylabel('Температура, °C');
title('Учебные измерения с пропуском');
```

5 Трёхмерные траектории: функция plot3

Функция plot3(x,y,z) строит пространственную линию по трём координатам. Длины массивов должны совпадать, поскольку каждый индекс соответствует одной точке траектории. Такая визуализация применяется в робототехнике, при исследовании движения БПЛА и в задачах пространственного мониторинга.

Для параметрической траектории x(t)=Rcos(t), y(t)=Rsin(t), z(t)=kt формируется винтовая линия. Параметр R задаёт радиус, а k — скорость изменения высоты относительно параметра t. Если t измеряется в радианах, аргументы sin и cos согласованы с MATLAB.

Команда axis equal задаёт одинаковый масштаб единиц по осям, что особенно важно для геометрических траекторий. Без неё окружность может визуально выглядеть эллипсом. Команда view(az,el) изменяет азимут и угол возвышения камеры.

В реальной навигации координаты могут измеряться в метрах, а временные метки храниться отдельно. На трёхмерном графике можно показать путь в пространстве, но оценку скорости и времени прохождения маршрута следует проводить дополнительно.

Если робот движется по плоскости, plot3 также может использоваться для отображения времени в качестве третьей оси. В таком случае ось z следует подписывать «Время, с», а не «Высота, м».

5

Трёхмерные траектории: функция plot3 (продолжение)

```
t = linspace(0,4*pi,300);
R = 2; k = 0.15;
x = R*cos(t); y = R*sin(t); z = k*t;
figure;
plot3(x,y,z,'LineWidth',1.8);
grid on; axis equal;
xlabel('X, м'); ylabel('Y, м'); zlabel('Z, м');
title('Пример пространственной траектории');
view(40,25);
```

6 Создание координатных сеток: meshgrid

Для построения поверхности z=f(x,y) нужно вычислить значение функции для множества пар координат. MATLAB предоставляет функцию meshgrid, которая из двух одномерных векторов создаёт матрицы координат X и Y.

Например, если x=[−1,0,1], y=[−2,0,2], то X содержит повторения строки x, а Y — повторения значений y по строкам. После этого матрица Z рассчитывается поэлементно: Z=X.^2+Y.^2.

Важное правило: если X и Y являются матрицами, то операции над ними при вычислении Z обычно записывают с точкой: .*, ./ и .^. Запись X^2 для произвольной прямоугольной матрицы не означает поэлементное возведение в квадрат и может привести к ошибке.

Плотность сетки определяет детализацию поверхности и объём вычислений. Сетка из 101×101 точек содержит 10 201 узел, а сетка из 1001×1001 — более миллиона. Избыточное разрешение увеличивает расход памяти без обязательного повышения научной информативности.

При построении поверхности необходимо указывать диапазон изменения каждой независимой переменной и проверять физический смысл модели на границах области.

6

Создание координатных сеток: meshgrid (продолжение)

```
x = -2:0.1:2;
y = -3:0.1:3;
[X,Y] = meshgrid(x,y);
Z = X.^2 + 0.5*Y.^2;
disp(size(X)); disp(size(Y)); disp(size(Z));
```

7 Каркасные поверхности: функция mesh

Функция mesh(X,Y,Z) отображает трёхмерную поверхность в виде сетки линий. Это удобно, когда необходимо видеть структуру дискретизации, геометрию модели или изменение функции по двум аргументам.

Рассмотрим учебную модель поверхности z=sin(x)cos(y). Значение z изменяется от −1 до 1, а форма поверхности определяется периодичностью тригонометрических функций. Для корректного вычисления по сетке необходимо использовать Z=sin(X).*cos(Y).

Каркасная визуализация полезна при сравнении результатов численного расчёта на разных сетках. Слишком крупная сетка может скрыть локальные особенности, а слишком мелкая создаёт визуальную перегруженность.

В научном отчёте каркасная поверхность должна сопровождаться пояснением, что изображают оси и что означает высота. При отображении результатов моделирования необходимо указывать параметры и границы области.

7 Каркасные поверхности: функция mesh (продолжение)

Функция mesh не выполняет интерполяцию физических измерений сама по себе. Если данные получены в нерегулярных точках, предварительно требуется корректная процедура построения сетки или интерполяции.

```
x = linspace(-pi,pi,60);
y = linspace(-pi,pi,60);
[X,Y] = meshgrid(x,y);
Z = sin(X).*cos(Y);
figure;
mesh(X,Y,Z);
grid on; xlabel('x'); ylabel('y'); zlabel('z');
title('Каркасная поверхность z = sin(x)cos(y)');
```

8 Цветные поверхности: функция surf

Функция surf(X,Y,Z) строит заполненную трёхмерную поверхность. Цвет обычно связан со значением высоты Z или переданными цветовыми данными. В отличие от mesh, она позволяет лучше воспринимать непрерывные изменения параметра на поверхности.

Команда colorbar добавляет шкалу соответствия цвета и численного значения. При визуализации температуры её необходимо снабжать физическим смыслом: цветовая шкала показывает температуру в градусах Цельсия, если именно температура задана в Z.

Команда shading interp сглаживает цветовые переходы между вершинами сетки, но не улучшает точность исходных данных. Она изменяет только способ отображения. Для научной интерпретации нельзя путать визуальное сглаживание с дополнительными измерениями.

Функция surf подходит для визуализации распределения температуры по поверхности, зависимости энергии от двух параметров, результатов оптимизации и ошибок управления. Если на осях отложены технологические параметры, следует избегать утверждений о физическом пространственном распределении.

При сравнении двух поверхностей важно использовать сопоставимые диапазоны цветов и осей. Автоматическое масштабирование может сделать одинаковые по величине различия визуально несопоставимыми.

8 Цветные поверхности: функция surf (продолжение)

```
x = linspace(-2,2,70);
y = linspace(-2,2,70);
[X,Y] = meshgrid(x,y);
Z = 35 + 8*exp(-(X.^2+Y.^2));
figure;
surf(X,Y,Z);
shading interp; colorbar;
xlabel('X, м'); ylabel('Y, м'); zlabel('Температура, °C');
title('Модель распределения температуры');
view(45,30);
```

9 Ракурс, цветовые карты и масштабы

В трёхмерных графиках выбор ракурса влияет на видимость особенностей поверхности. Команда view(2) переключает вид сверху, а view(3) — стандартный пространственный вид. Для анализа локальных максимумов полезно сопоставлять пространственную поверхность и проекцию сверху.

Команды xlim, ylim, zlim устанавливают границы осей. Функция clim задаёт диапазон цветовой шкалы в современных версиях MATLAB; при использовании старых версий может применяться caxis. При сравнении разных наборов данных границы цветовой шкалы желательно фиксировать.

Функция colormap выбирает цветовую карту. Выбор должен обеспечивать читаемость и однозначное восприятие последовательных значений. Не следует использовать случайную палитру, если она создаёт ложные границы в непрерывном распределении.

Визуальные параметры — толщина линий, размер маркеров, размер шрифта, расположение легенды — должны обеспечивать читаемость после вставки изображения в Word или презентацию. График, хорошо читаемый на большом мониторе, может оказаться непригодным в печатном отчёте.

Для воспроизводимости рисунка рекомендуется сохранять исходные данные, скрипт, параметры модели и команду экспорта. Номер рисунка и его подпись оформляются уже в учебном документе или научной статье.

10 Прикладной пример: двумерный мониторинг температуры IoT

Постановка задачи: IoT-узел передаёт последовательность значений температуры каждые 10 секунд. Необходимо построить временной график, обозначить порог 28 °C, вычислить среднее и максимальное значение и показать превышения. Для первого этапа используются учебные данные, чтобы не зависеть от доступности физического датчика.

Пусть t=[0,10,...,100] c, а T=[25.0,25.3,25.8,26.1,26.7,27.0,27.4,27.8,28.1,28.5,29.0] °C. MATLAB строит линию с маркерами, горизонтальный порог и выводит сводные показатели. Превышения определяются логическим выражением T>28.

Проверка корректности включает сравнение числа временных меток и измерений, контроль NaN/Inf и единиц измерения. Для реального проекта необходимо дополнительно сохранять идентификатор датчика и время получения данных.

Этот пример демонстрирует переход от массива чисел к инженерной информации. Важен не только график, но и вывод: в последних трёх точках температура превышает выбранный учебный порог.

10

Прикладной пример: двумерный мониторинг температуры IoT (продолжение)

```
t = 0:10:100;
T = [25 25.3 25.8 26.1 26.7 27 27.4 27.8 28.1 28.5 29];
assert(numel(t)==numel(T));
assert(all(isfinite(T)));
limit = 28;
figure;
plot(t,T,'-o','LineWidth',1.5); hold on;
yline(limit,'--','Порог');
grid on; xlabel('Время, с'); ylabel('Температура, °C');
title('Мониторинг температуры: учебные данные');
fprintf('Среднее %.2f °C; максимум %.2f °C
',mean(T),max(T));
fprintf('Превышений: %d
',sum(T>limit));
```

11

Прикладной пример: трёхмерная карта технологического параметра

Рассмотрим учебную модель распределения температуры по прямоугольной поверхности. Пусть X и Y — координаты в метрах, а температура определяется формулой $T(X,Y)=25+15\exp(-((X-1)^2+(Y+0,5)^2)/2)$. Такая функция задаёт локальную нагретую область около точки (1;−0,5).

Сначала создают сетку meshgrid, затем вычисляют матрицу T. Для наглядности строят рядом каркасную поверхность mesh и заполненную поверхность surf. По цветовой шкале и высоте можно определить положение максимума.

Модель имеет иллюстративный характер. Реальное температурное поле требует измерений, определения граничных условий или обоснованной физической модели. Нельзя утверждать, что полученная поверхность является экспериментальной картой без исходных измерений.

Сравнение mesh и surf показывает их различную визуальную роль: первая подчёркивает структуру сетки, вторая — распределение величины. При подготовке отчёта достаточно выбрать наиболее информативный вариант либо использовать оба с разными пояснениями.

11

Прикладной пример: трёхмерная карта технологического параметра (продолжение)

```
x = -3:0.1:3; y = -3:0.1:3;
[X,Y] = meshgrid(x,y);
T = 25 + 15*exp(-((X-1).^2+(Y+0.5).^2)/2);
figure;
tiledlayout(1,2);
nexttile; mesh(X,Y,T); grid on;
xlabel('X, м'); ylabel('Y, м'); zlabel('T, °C');
title('mesh: каркас');
nexttile; surf(X,Y,T); shading interp; colorbar;
xlabel('X, м'); ylabel('Y, м'); zlabel('T, °C');
title('surf: поверхность');
```

12

Требования к научным графикам и типичные ошибки

Качественный график должен содержать понятные оси, единицы измерения, подпись, достаточную контрастность, адекватный масштаб и обозначение источника данных. Если представлены результаты математической модели, это должно быть прямо указано.

Распространённая ошибка №1 — использование plot3 при несовпадающих длинах x, y и z. Необходимо проверить numel каждого массива и соответствие координат одной точке.

Ошибка №2 — использование матричных операторов вместо поэлементных при вычислении Z. Для сеток X и Y обычно применяются .^, .* и ./, иначе выражение может завершиться ошибкой либо дать другой математический результат.

Ошибка №3 — построение поверхности по нерегулярным измерениям без корректной интерполяции. Функции mesh и surf предполагают согласованную сетку или совместимые массивы координат; произвольная перестановка измерений в матрицу не создаёт физически обоснованной поверхности.

Ошибка №4 — отсутствие цветовой шкалы и единиц измерения. В результате читатель не может определить, что означают цвета и значения высоты.

12

Требования к научным графикам и типичные ошибки (продолжение)

Ошибка №5 — чрезмерное сглаживание или скрытие пропусков. Визуальная плавность не является доказательством качества данных. При публикации важно сохранять информацию о дискретности измерений.

Ошибка №6 — выбор масштаба, который скрывает важные отклонения. При сравнении вариантов следует по возможности использовать общие пределы осей и цвета.

Ошибка №7 — использование разных цветов без легенды. Если на одном графике показано несколько сигналов, их обозначения должны быть однозначными и соответствовать тексту отчёта.

3 Заключение

В лекции изучены методы визуализации двумерных и трёхмерных данных в MATLAB Online. Рассмотрены функции plot, plot3, meshgrid, mesh и surf, правила оформления графиков и выбора цветовой шкалы. На учебных примерах показано построение временного ряда температуры, пространственной траектории и поверхности распределения технологического параметра. Освоенные инструменты необходимы для последующих задач интерполяции, экстраполяции, моделирования и анализа данных IoT.

Контрольные вопросы (1/2)

1. Какова роль графической визуализации в интеллектуальных системах управления?

2. Чем экспериментальный график отличается от графика математической модели?

3. Каково назначение функции plot?

4. Какие требования предъявляются к векторам x и y в plot(x,y)?

5. Для чего применяются figure и grid on?

6. Как задать подписи осей и заголовок графика?

7. Как вывести две линии на одной системе координат?

8. Для чего используется legend?

9. Как обозначить пороговое значение на графике?

10. Чем отличаются plot, scatter и stem?

11. Как MATLAB отображает пропуски NaN в линии?

12. Для чего предназначена функция plot3?

13. Какие массивы необходимы для трёхмерной траектории?

14. Почему при построении геометрических траекторий полезна axis equal?

15. Как изменить ракурс трёхмерного графика?

16. Для чего предназначена функция meshgrid?

Контрольные вопросы (2/2)

17. Почему при вычислении Z по сетке нужны поэлементные операторы?

18. Каково назначение функции mesh?

19. Чем функция surf отличается от mesh?

20. Для чего используется colorbar?

21. Что изменяет команда shading interp?

22. Как установить диапазон осей и цветовой шкалы?

23. Какие ошибки возникают при неверной интерпретации 3D-поверхности?

24. Как подготовить график MATLAB для научного отчёта?

25. Какие графики подходят для мониторинга температуры и анализа движения робота?

Д Задания для самостоятельного закрепления

1. Постройте графики sin(x) и cos(x) на одном рисунке при x от 0 до 2π; оформите легенду и оси.

2. Постройте три кривые охлаждения с τ=60, 100 и 160 с. Объясните влияние постоянной времени.

3. Постройте пространственную траекторию робота по формулам x=2cos(t), y=2sin(t), z=0,1t при t от 0 до 6π.

4. Сформируйте сетку X,Y и постройте mesh для Z=sin(X).*cos(Y) на интервале [−π,π].

5. Для Z=30+10exp(−X.^2−Y.^2) постройте surf с colorbar и видом сверху.

6. По учебным измерениям температуры постройте график с порогом 28 °C и выделите превышения.

Рекомендуемые источники

- Исходный силлабус «Проект MATLAB IoT», тема лекции №4: средства MATLAB для построения двух- и трёхмерных графиков с помощью plot3, surf и mesh.
- MathWorks. MATLAB Documentation: https://www.mathworks.com/help/matlab/
- MathWorks. 2-D and 3-D Plots: https://www.mathworks.com/help/matlab/2-and-3d-plots.html
- MathWorks. plot, plot3, mesh, surf и meshgrid — справочные страницы соответствующих функций.