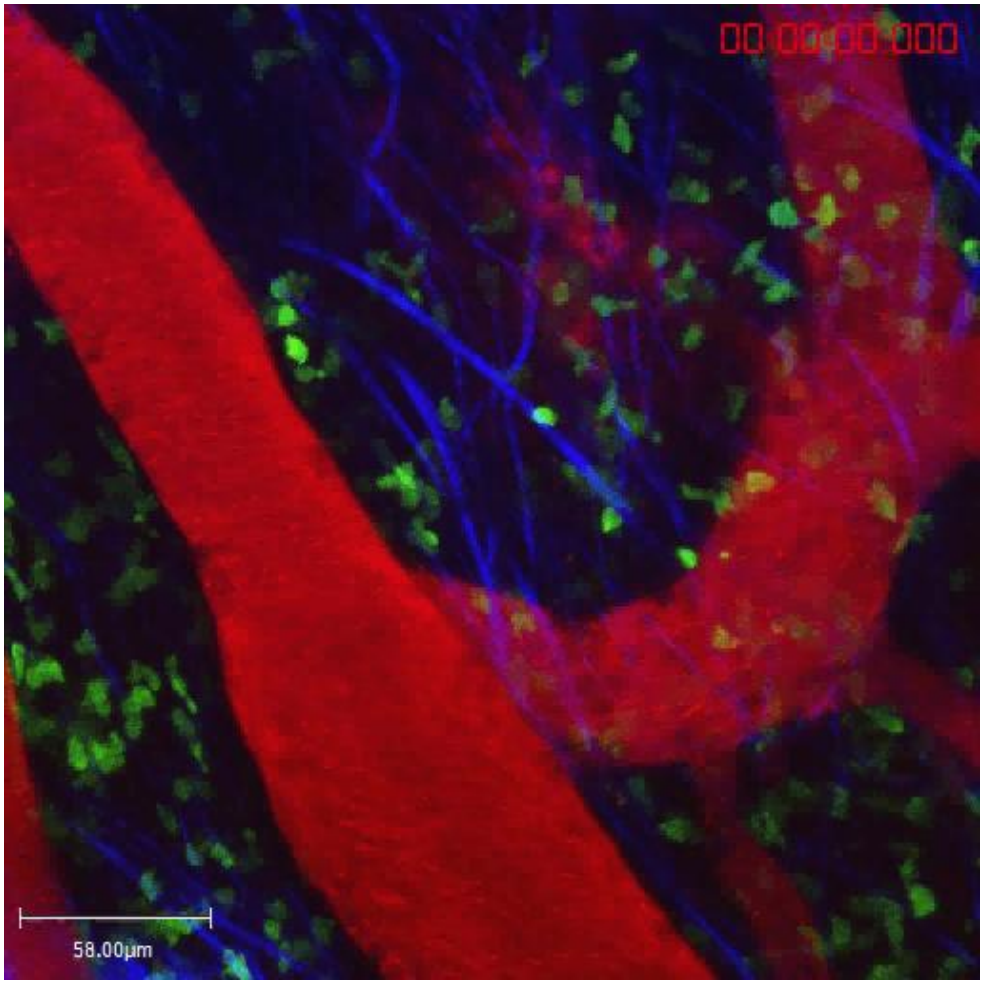
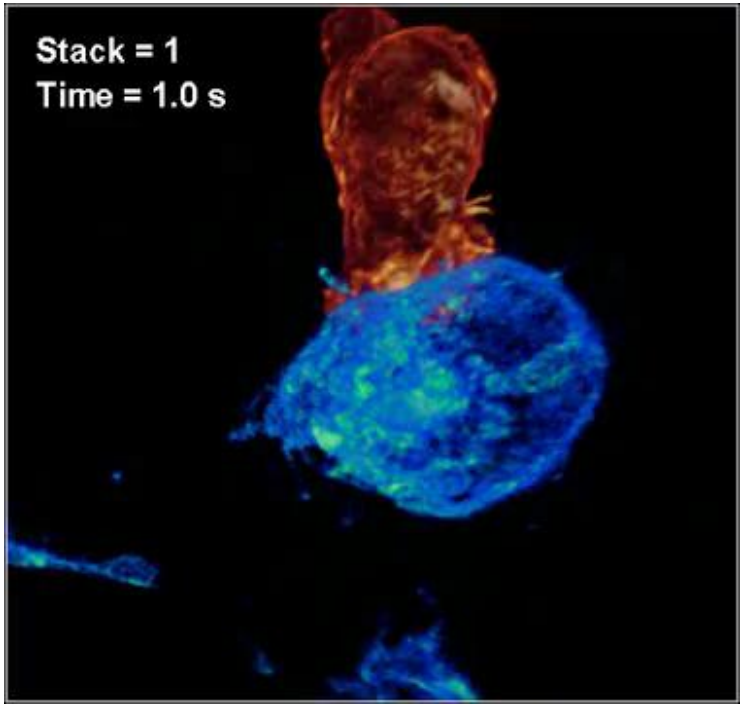
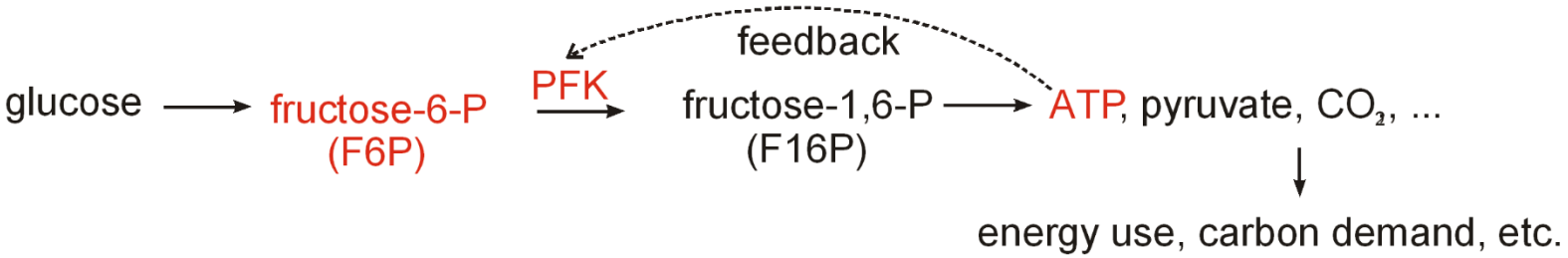
Dynamic systems drive cellular functions





Dynamic systems – HW2Q1 as starting point

In the glycolytic pathway, one point of regulation is the conversion of fructose -6-P to fructose -1,6-P by the enzyme phosphofructokinase (PFK). ATP, a downstream product of the glycolysis pathway, acts to inhibit PFK at high concentrations, slowing down the generation of ATP and regulating the overall level in the cell. Assume ATP acts as an allosteric, non -competitive inhibitor. The components that are central to this question are marked in red.

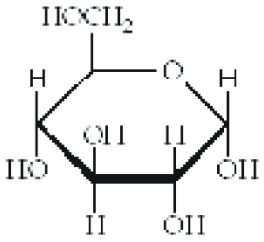


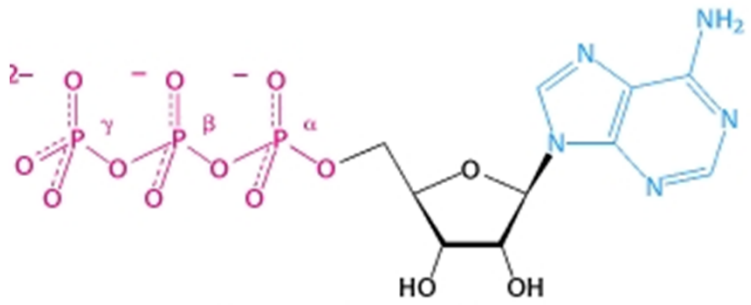
Glycolytic oscillations – Sel'kov model

Glucose → Fructose 6-P → Fructose 1,6-P → Pyruvate

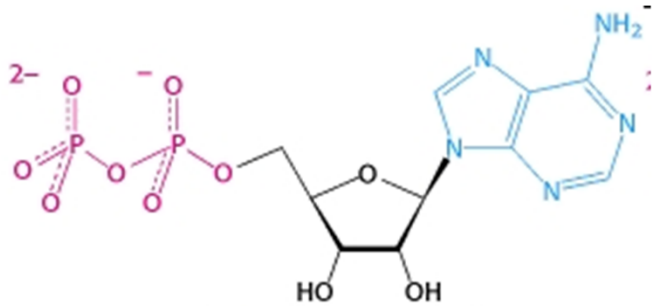
PFK1

ATP ADP

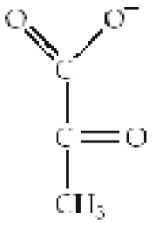




Adenosine triphosphate (ATP)

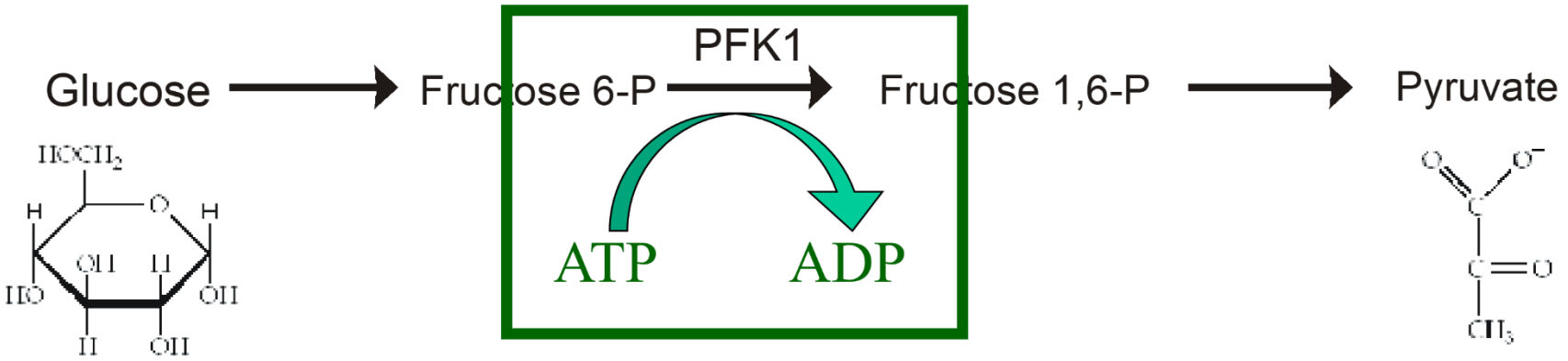


Adenosine diphosphate (ADP)



ATP production

Glycolytic oscillations – Sel'kov model

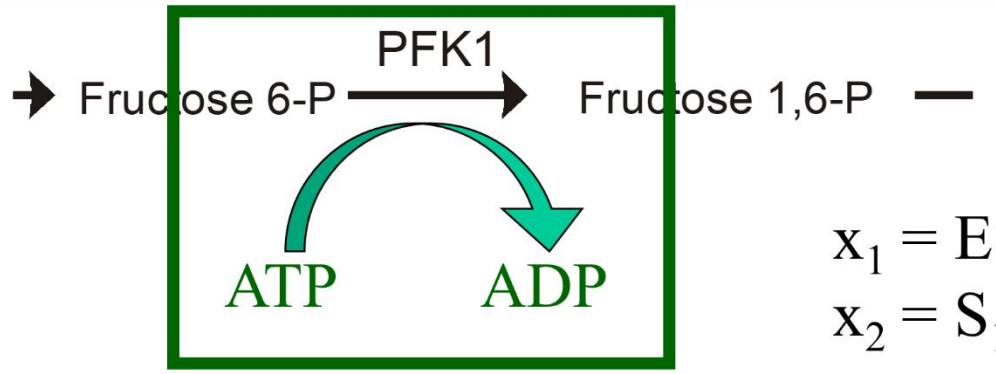


- Steady state doesn't capture full behavior
- Oscillations have been observed in experimental systems
- **Sel'kov** model as a simplified explanation

Competing feedback loops

- ATP is both a substrate and inhibitor of PFK1 (negative feedback)
  - Focus here on ATP as a substrate, to be acted upon by PFK1
- Binding of multiple ADPs activates PFK1 (positive feedback)
  - increasing amounts of ADP enhance reaction rate
  - acts through increasing AMP concentration (won't include that here)
- Goal: find reaction behavior as a function of input ATP

Glycolytic oscillations – Sel'kov model



E = PFK1
S_1=ATP
S_2=ADP

$x_1 = ES_2^\gamma$
$x_2 = S_1 ES_2^\gamma$

Activation of PFK1 by ADP
requires γ ADPs for activation

$$\gamma S_2 + E \underset{k_{-3}}{\overset{k_3}{\rightleftharpoons}} ES_2^\gamma,$$

PFK1 pathways

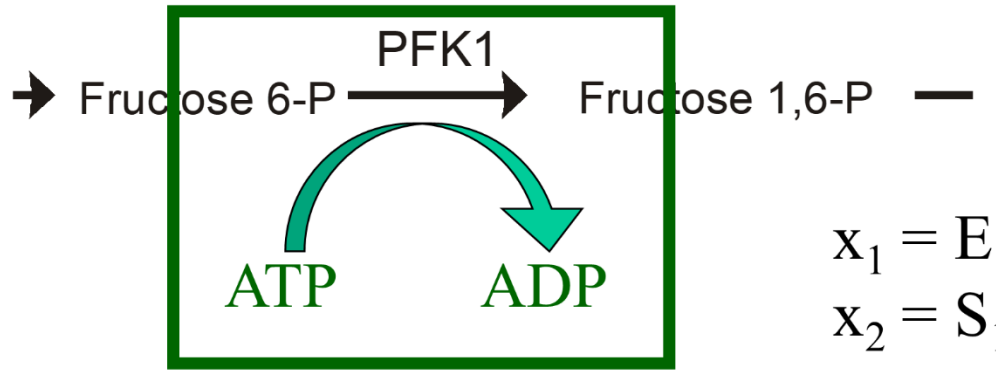
$$\xrightarrow{v_1} S_1,$$

$$S_1 + ES_2^\gamma \underset{k_{-1}}{\overset{k_1}{\rightleftharpoons}} S_1 ES_2^\gamma \xrightarrow{k_2} ES_2^\gamma + S_2$$

$$S_2 \xrightarrow{v_2}.$$

Given input rate v_1, how does this system behave?

Reaction scheme



E = PFK1
S_1=ATP
S_2=ADP

$x_1 = ES_2^{\gamma}$
$x_2 = S_1 ES_2^{\gamma}$

$$\frac{ds_1}{dt} = v_1 - k_1 s_1 x_1 + k_{-1} x_2,$$

$$\frac{ds_2}{dt} = k_2 x_2 - k_3 s_2^{\gamma} e + k_{-3} x_1 - v_2 s_2,$$

$$\frac{dx_1}{dt} = -k_1 s_1 x_1 + (k_{-1} + k_2) x_2 + k_3 s_2^{\gamma} e - k_{-3} x_1,$$

$$\frac{dx_2}{dt} = k_1 s_1 x_1 - (k_{-1} + k_2) x_2.$$

$$e + x_1 + x_2 = e_0$$

$$\sigma_1 = \frac{k_1 s_1}{k_2 + k_{-1}} \qquad u_1 = \frac{x_1}{e_0} \qquad t = \frac{k_2 + k_{-1}}{e_0 k_1 k_2}\tau$$

$$\sigma_2 = \left(\frac{k_3}{k_{-3}}\right)^{1/\gamma} s_2 \qquad u_2 = \frac{x_2}{e_0} \qquad \nu = \frac{v_1}{k_2 e_0}$$

$$\varepsilon = \frac{e_0 k_1 k_2}{(k_2 + k_{-1})^2} \qquad \eta = \frac{v_2 (k_2 + k_{-1})}{k_1 k_2 e_0} \qquad \alpha = \frac{k_2 + k_{-1}}{k_1}\left(\frac{k_3}{k_{-3}}\right)^{1/\gamma}$$

$$\frac{d\sigma_1}{d\tau} = \nu - \frac{k_2 + k_{-1}}{k_2} u_1 \sigma_1 + \frac{k_{-1}}{k_2} u_2,$$

$$\frac{d\sigma_2}{d\tau} = \alpha \left[u_2 - \frac{k_{-3}}{k_2} \sigma_2^{\gamma} (1 - u_1 - u_2) + \frac{k_{-3}}{k_2} u_1 \right] - \eta \sigma_2,$$

$$0 = \epsilon \frac{du_1}{d\tau} = u_2 - \sigma_1 u_1 + \frac{k_{-3}}{k_2 + k_{-1}} \left[\sigma_2^{\gamma} (1 - u_1 - u_2) - u_1 \right],$$

$$0 = \epsilon \frac{du_2}{d\tau} = \sigma_1 u_1 - u_2,$$

time-scaling (quasi-steady state)

$$u_1 = \frac{\sigma_2^\gamma}{\sigma_2^\gamma \sigma_1 + \sigma_2^\gamma + 1},$$

$$u_2 = \frac{\sigma_1 \sigma_2^\gamma}{\sigma_2^\gamma \sigma_1 + \sigma_2^\gamma + 1} = f(\sigma_1, \sigma_2),$$

which allows simplification of the sigma differential equations:

$$\frac{d\sigma_1}{d\tau} = \nu - f(\sigma_1, \sigma_2),$$

$$\frac{d\sigma_2}{d\tau} = \alpha f(\sigma_1, \sigma_2) - \eta \sigma_2.$$

Give some specific numbers:

Setting: $\gamma=2$; $\nu=0.0285$; $\eta=0.1$; $\alpha=1.0$

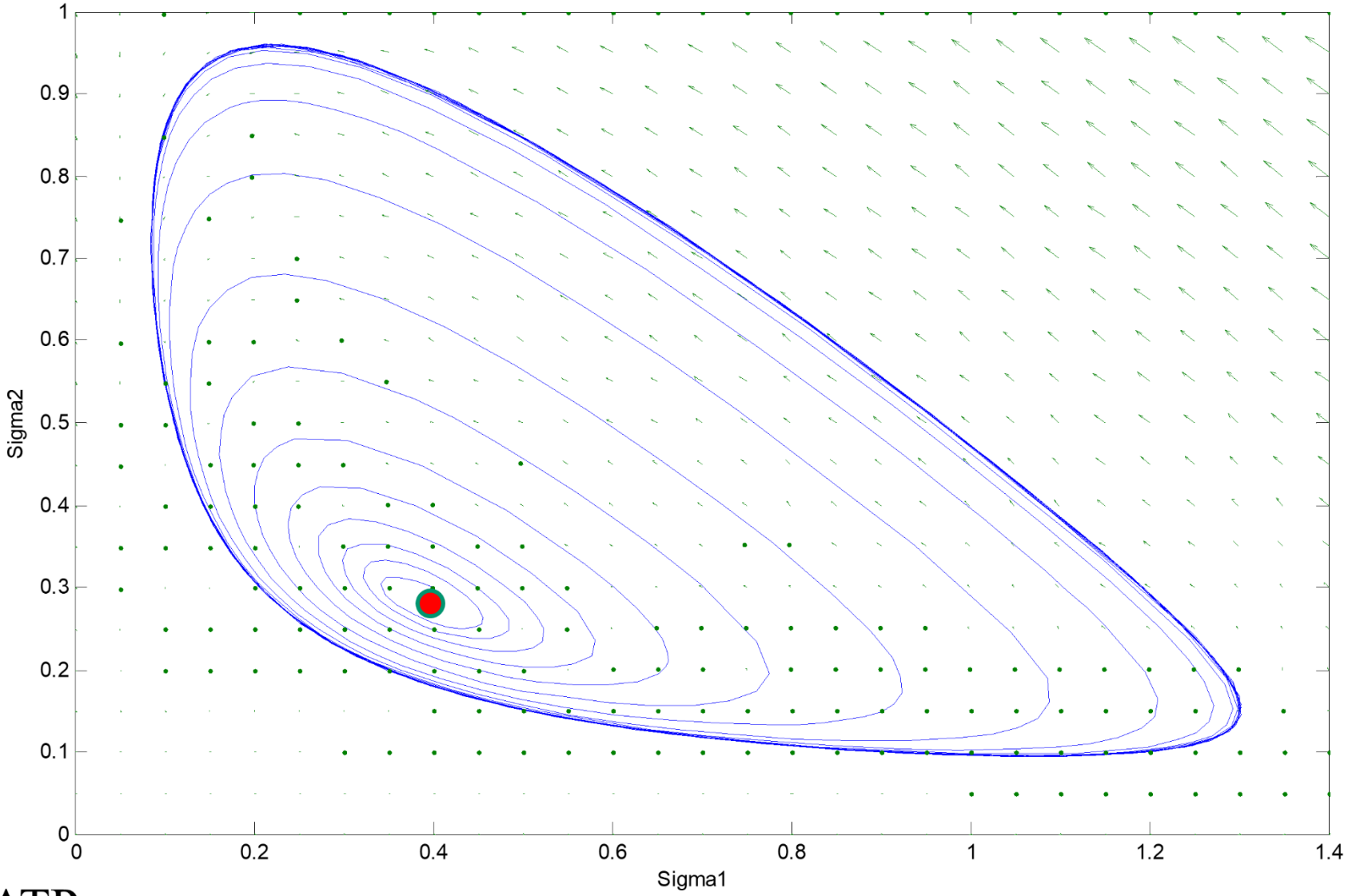
Stable solution $\sigma_1= 0.3905$; $\sigma_2= 0.285$

which allows simplification of the sigma differential equations:

$$\frac{d\sigma_1}{d\tau} = \nu - f(\sigma_1, \sigma_2),$$

$$\frac{d\sigma_2}{d\tau} = \alpha f(\sigma_1, \sigma_2) - \eta\sigma_2.$$

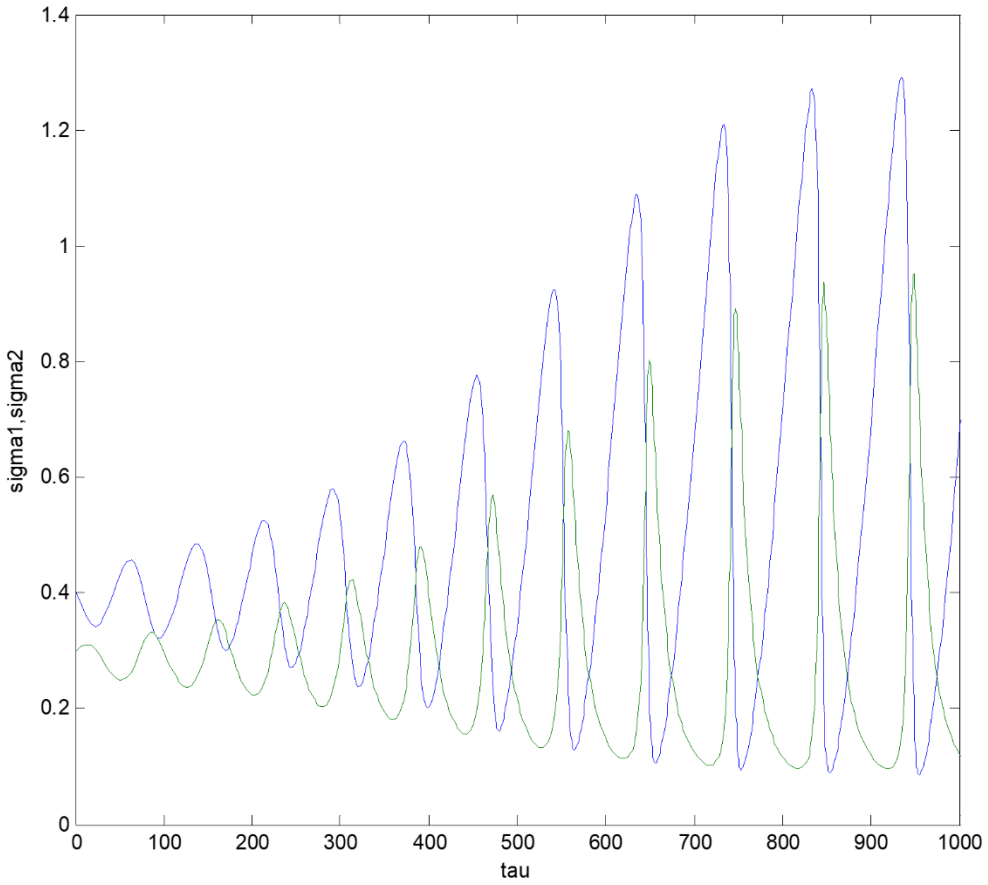
vector map



S_1=ATP
S_2=ADP

γ=2; ν=0.0285; η=0.1; α=1.0

temporal dynamics



S_1=ATP

S_2=ADP

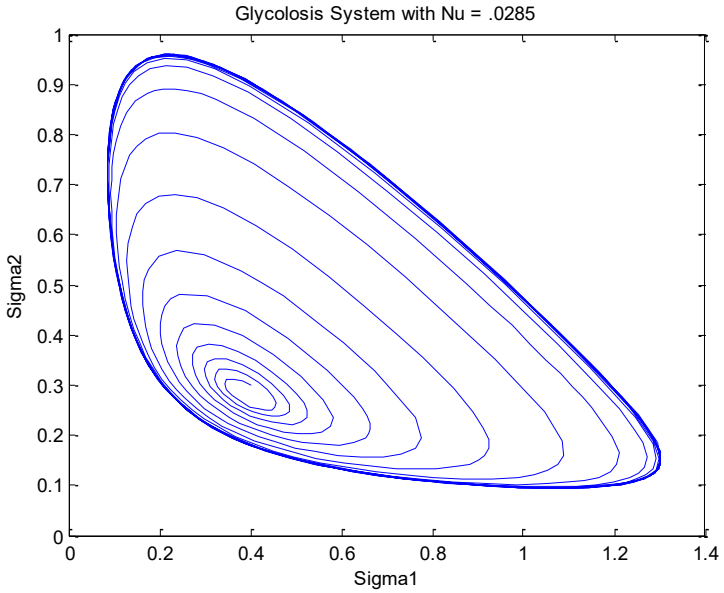
$$\xrightarrow{v_1} S_1,$$

$$\gamma S_2 + E \underset{k_{-3}}{\overset{k_3}{\rightleftharpoons}} ES_2^{\gamma},$$

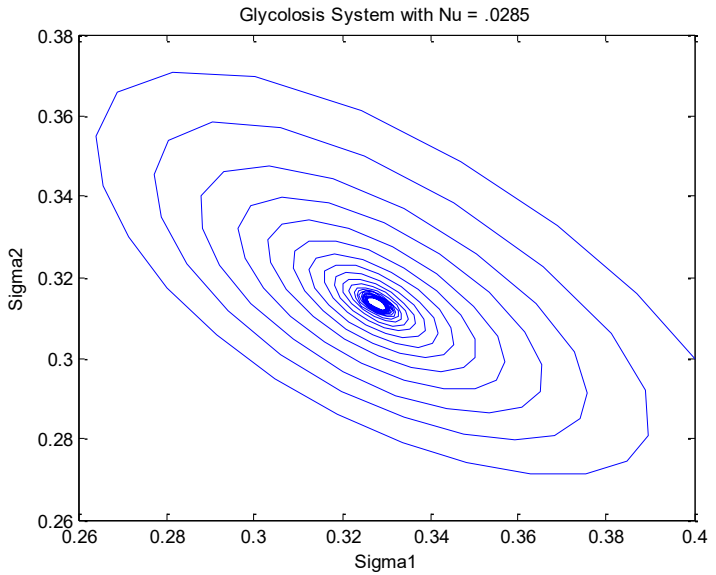
$$S_1 + ES_2^{\gamma} \underset{k_{-1}}{\overset{k_1}{\rightleftharpoons}} S_1ES_2^{\gamma} \xrightarrow{k_2} ES_2^{\gamma} + S_2$$

$$S_2 \xrightarrow{v_2}.$$

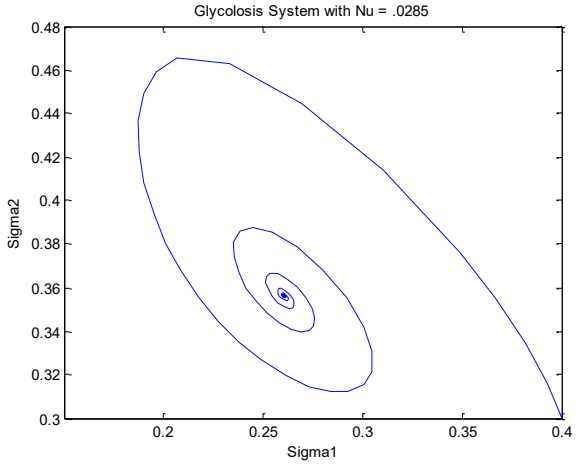
Oscillations don’t occur for all parameters



$\alpha = 1$



$\alpha = 1.1$



$\alpha = 1.25$

MATLAB

- Good, general-purpose mathematics package.
- Great for real data, so-so on symbolic
- Based on the manipulation and processing of matrices / lists (arrays)
- Full programming language
- Good support both from The Mathworks and general users
- Google and the help screens are a great resource

MATLAB vs. Python

MATLAB	Python
• Commercial	• Open source
• Designed for real data	• Adaptable
• Toolboxes for specialization	• Many libraries. For math: NumPy, SciPy
• Core programming capabilities	• “elegant” programming syntax
• Inefficient execution	• Inefficient execution

•https://cuit.columbia.edu/content/matlab

MATLAB

MATLAB is a high-level technical computing language and interactive environment for algorithm development, data visualization, data analysis, and numeric computation.

MATLAB Portal Login

Make a request

Please click the "Make a request" button to obtain a software license.

Some departments and schools designate a liaison to act as a single point of contact for everyone within their organization to order licenses, install the software, and in some cases, manage a lab server for that department.

If you don't see your school or department on the MATLAB liaison list, please ask your departmental or financial administrator, immediate supervisor, or department head to submit a ticket to request individual license orders, bulk orders and all renewals. Please send an ARC chartstring along with the name and email address of the person(s) needing the license.

You can use MATLAB in a wide range of applications, including signal and image processing, communications, control design, test and measurement, financial modeling and analysis, and computational biology. Add-on toolboxes extend the Matlab environment to solve particular classes of problems in these application areas. View a full list of MATLAB products and applications available via Columbia's license.

RELATED INFORMATION

- CU's MATLAB Product Set
- MATLAB online training courses
- Mathematica
- SAS
- SPSS
- Site License Distribution Policy

Customers purchasing MATLAB are responsible for the installation and correct use of the product, as well as understanding hardware and operating system requirements. For costs, please see the MATLAB pricing section on this page. CUIT can create MATLAB server licenses to be managed by your departmental staff. Annual license renewals, ISO images, and new release updates are provided upon request.

Pricing for July 1, 2025 - June 30, 2026 validation period

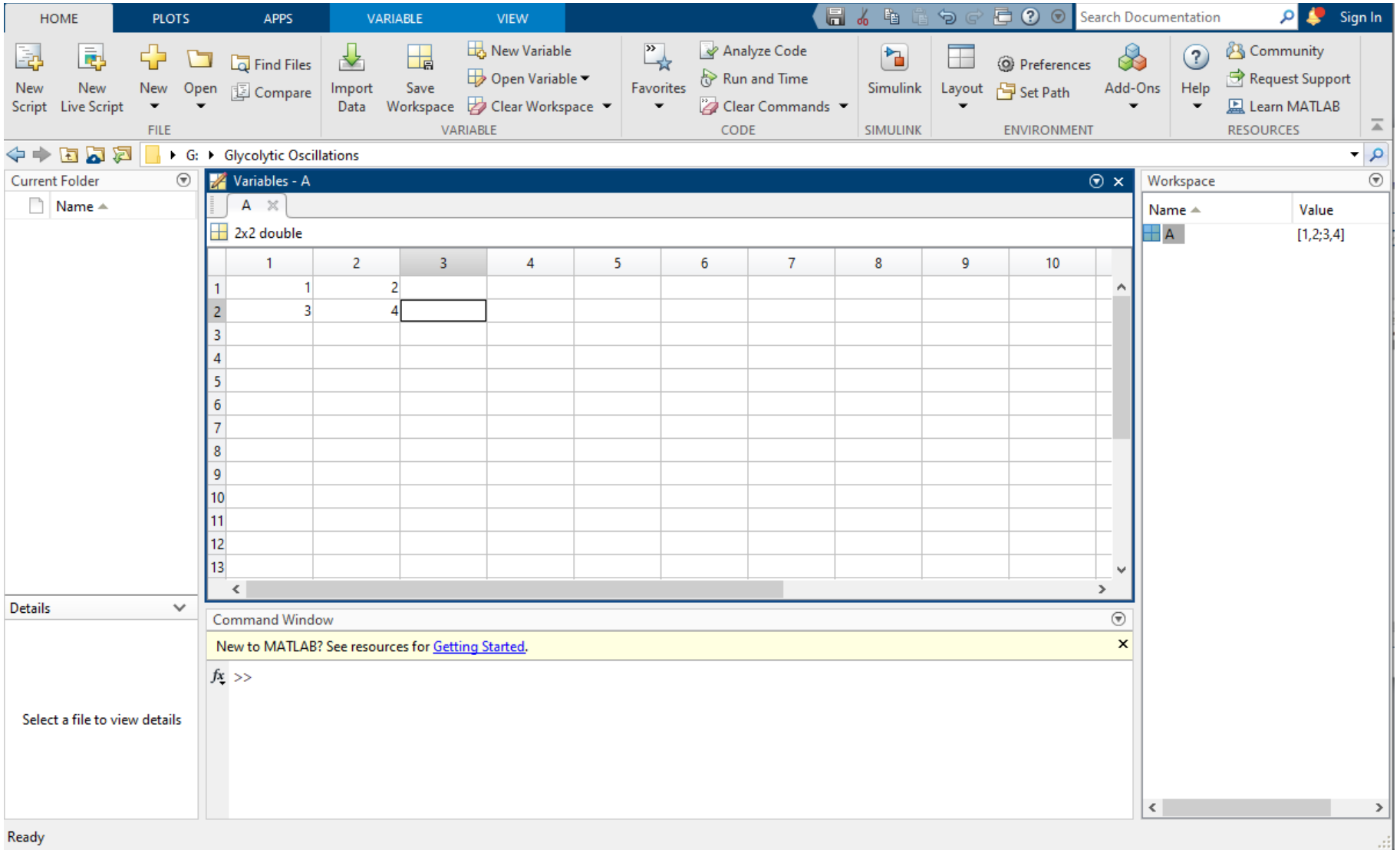
- The price is fixed and cannot be pro-rated.
- One license is the equivalent of either one computer (laptop, desktop) or a **single** Unix processor.
- Licenses are valid for a 1 year period: July 1 through June 30. Software **will not run** after the license expires.
- To get the best possible results and avoid any risk of downtime, please be sure to purchase a renewal at least **one month** before your license expires.

License Type	Price
Student licenses	No Charge
Individual campus licenses	$392

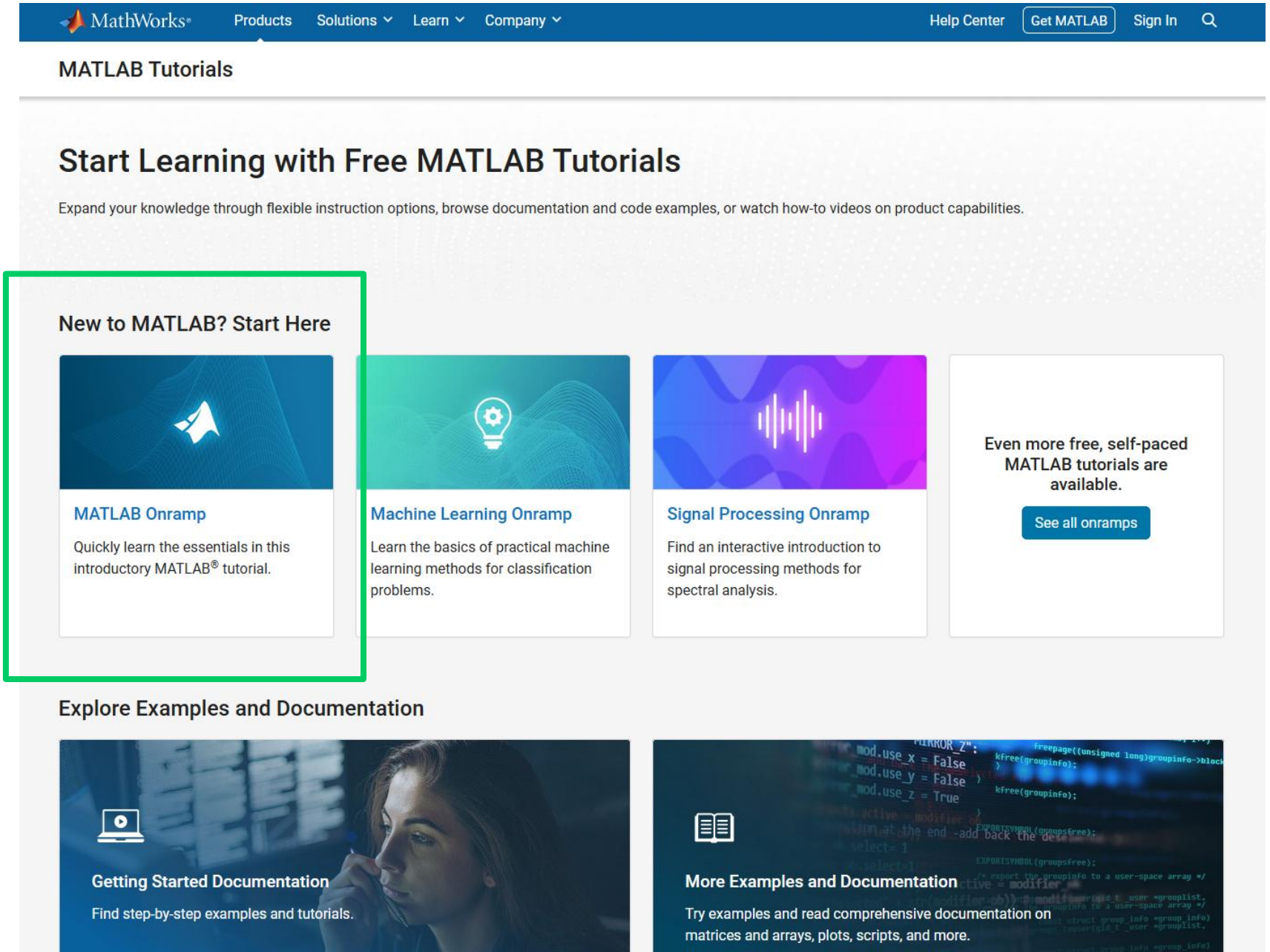
Support Contact

Documentation is available at MathWorks. Technical support is available through Mathworks Support, and online training courses are also available. Guides and other reference materials can also be purchased separately through third parties.

MATLAB


HOME
PLOTS
APPS
VARIABLE
VIEW
Search Documentation
Sign In
New Script
New Live Script
New
Open
Find Files
Compare
FILE
Import Data
Save Workspace
New Variable
Open Variable
Clear Workspace
VARIABLE
Favorites
Analyze Code
Run and Time
Clear Commands
CODE
Simulink
SIMULINK
Layout
Preferences
Set Path
Add-Ons
ENVIRONMENT
Help
Community
Request Support
Learn MATLAB
RESOURCES
G: ▸ Glycolytic Oscillations
Current Folder
Name
Variables - A
A
2x2 double
1 2 3 4 5 6 7 8 9 10
1 1 2
2 3 4
3 4 5 6 7 8 9 10 11 12 13
Workspace
Name
Value
A
[1,2;3,4]
Details
Select a file to view details
Command Window
New to MATLAB? See resources for Getting Started.
fx >>
Ready

- http://www.mathworks.com/support/learn-with-matlab-tutorials.html



MATLAB in this course will use:

- a basic knowledge of navigating and working in MATLAB.
- ODE solver, ode45
  - requires
    - set of ODEs describing system
    - initial conditions
    - timeframe of solution
  - there are other solvers available, but this is a good starting point

Arrays and Matrices

Matrices: m (rows) x n (column)

Entered as follows:

```
mat1=[1 2 3 ; 4 5 6]
```

generates the 2 x 3 array

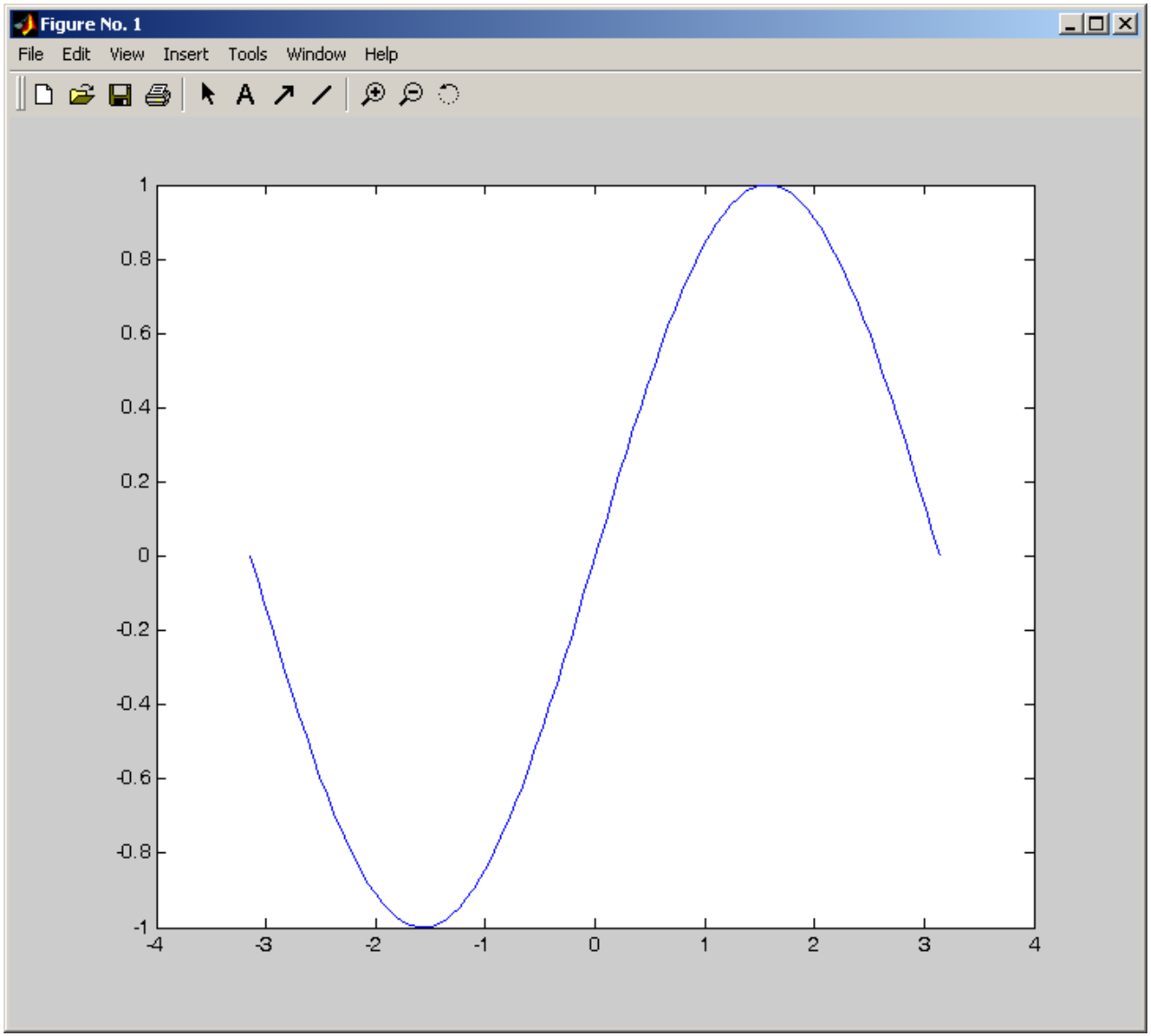
```
mat1=
     [1 2 3]
     [4 5 6]
```

Accessing elements:

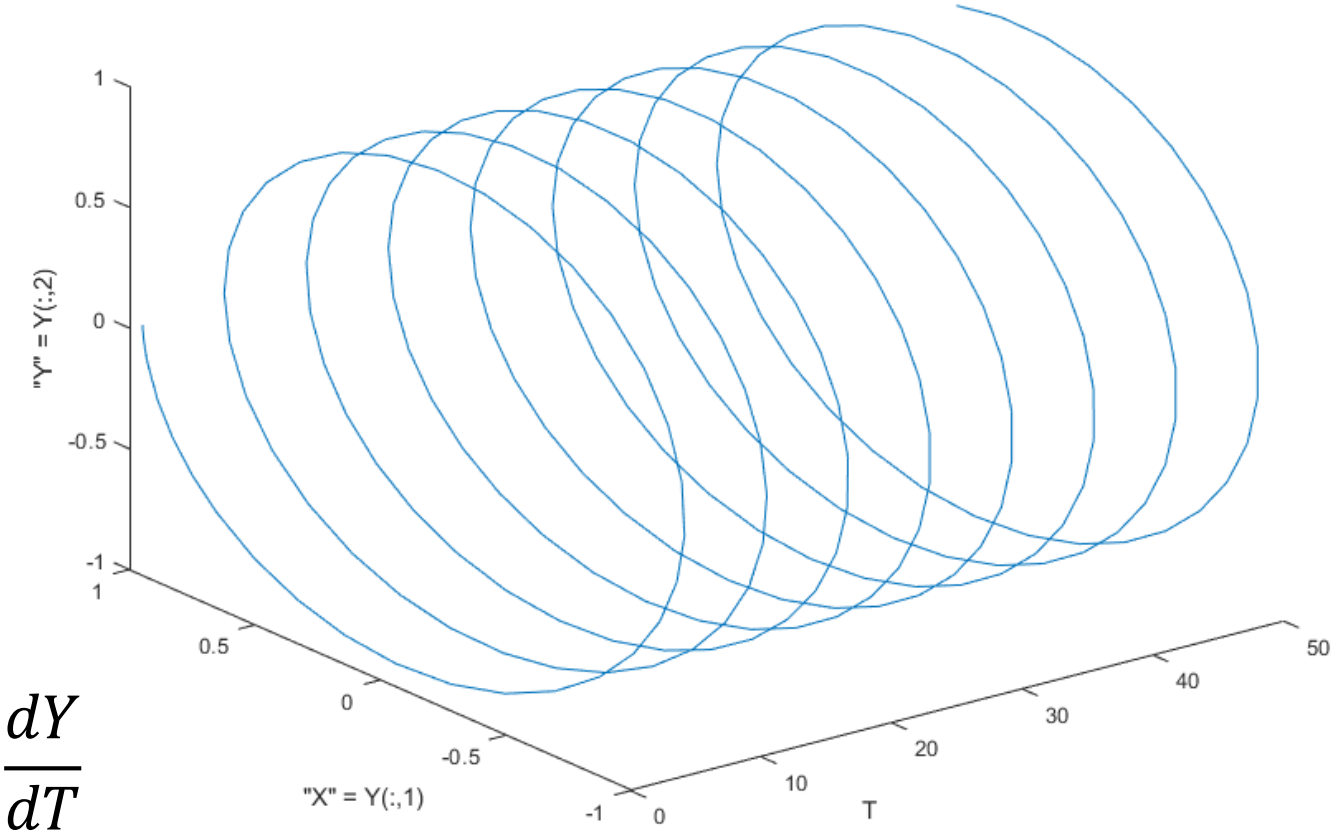
```
mat1(2,1) returns the value 4
mat1(2,:) returns the value [4 5 6]
```

Plotting graphs example

```
x=linspace(-pi,pi,100);
y=sin(x);
plot(x,y);
```



ODE example - spiral



$$\frac{dX}{dT} = \frac{dY}{dT}$$

$$\frac{dY}{dT} = -\frac{dX}{dT}$$

ODE example; sine and cosine

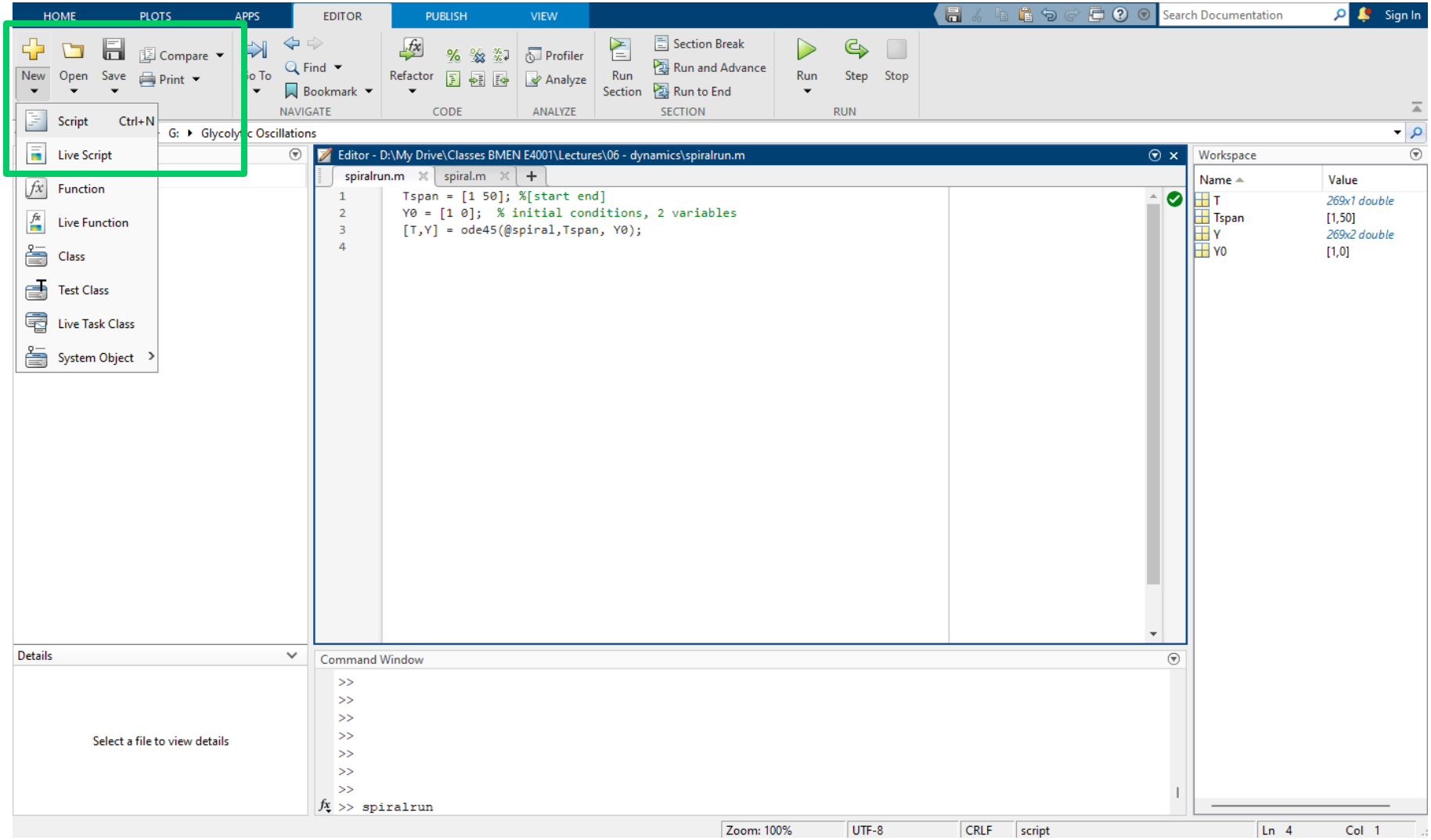
(file "spiralrun.m")

```
Tspan = [1 50]; %[start end]
Y0 = [1 0];  % initial conditions, 2 variables
[T,Y] = ode45(@spiral,Tspan, Y0);
```

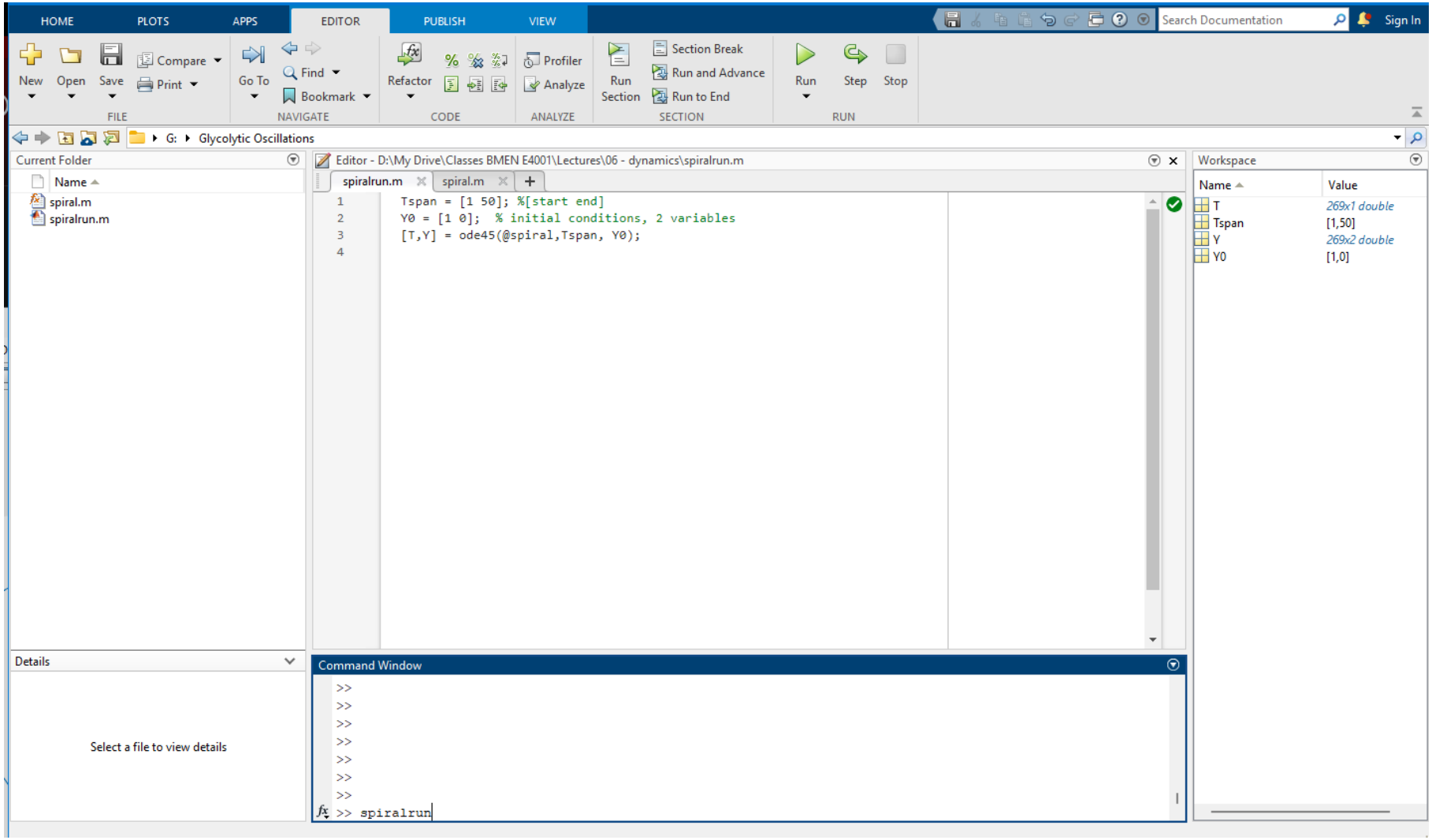
(file "spiral.m")

```
function dy = spiral(t,y)
dy=zeros(2,1); % column vector w/ 2 rows
dy(1)=y(2);
dy(2)=-y(1);
```

MATLAB


HOME
PLOTS
APPS
EDITOR
PUBLISH
VIEW
Search Documentation
Sign In
New
Open
Save
Compare
Print
Find
Bookmark
NAVIGATE
Refactor
CODE
Profiler
Analyze
ANALYZE
Run Section
Section Break
Run and Advance
Run to End
SECTION
Run
Step
Stop
RUN
Script Ctrl+N
Live Script
Function
Live Function
Class
Test Class
Live Task Class
System Object
Editor - D:\My Drive\Classes BMEN E4001\Lectures\06 - dynamics\spiralrun.m
spiralrun.m
spiral.m
1 Tspan = [1 50]; %[start end]
2 Y0 = [1 0]; % initial conditions, 2 variables
3 [T,Y] = ode45(@spiral,Tspan, Y0);
4
Workspace
Name
Value
T 269x1 double
Tspan [1,50]
Y 269x2 double
Y0 [1,0]
Details
Select a file to view details
Command Window
>> spiralrun
Zoom: 100%
UTF-8
CRLF
script
Ln 4
Col 1

MATLAB


HOME
PLOTS
APPS
EDITOR
PUBLISH
VIEW
Search Documentation
Sign In
New
Open
Save
Compare
Print
FILE
Go To
Find
Bookmark
NAVIGATE
Refactor
CODE
Profiler
Analyze
ANALYZE
Run Section
Section Break
Run and Advance
Run to End
SECTION
Run
Step
Stop
RUN
G: ▸ Glycolytic Oscillations
Current Folder
Name
spiral.m
spiralrun.m
Editor - D:\My Drive\Classes BMEN E4001\Lectures\06 - dynamics\spiralrun.m
spiralrun.m
spiral.m
1 Tspan = [1 50]; %[start end]
2 Y0 = [1 0]; % initial conditions, 2 variables
3 [T,Y] = ode45(@spiral,Tspan, Y0);
4
Workspace
Name
Value
T
269x1 double
Tspan
[1,50]
V
269x2 double
Y0
[1,0]
Details
Select a file to view details
Command Window
>>
>>
>>
>>
>>
>>
>>
>> spiralrun

ODE example; sine and cosine

(file "spiralrun.m")

```
Tspan = [1 50]; %[start end]
Y0 = [1 0];  % initial conditions, 2 variables
[T,Y] = ode45(@spiral,Tspan, Y0);
```

(file "spiral.m")

```
function dy = spiral(t,y)
dy=zeros(2,1); % column vector w/ 2 rows
dy(1)=y(2);
dy(2)=-y(1);
```

ODE solvers (ode45, ode23, ode113, ode15s, ode23s, ode23t, ode23tb)

Basic syntax:

[T,Y] = solver(odefun,tspan,y0)

where:

odefun = handle to user-defined function of form y' = f(t,y)

tspan = 2-element vector specifying beginning and ending tpoints

y0 = initial conditions (scalar or vector)

T = output vector, returns timepoints of solution

Y = output matrix, returns data of solution, 1 row corresponding to one value of associated T.

Structure of a function

function [out1, out2] = myfunc(in1, in2, in3)

1. Takes a number of input variabes (in1, in2, etc.)
2. Returns a set of output variables (out1, out2, etc.)
3. Simplest implementation, save as a local M-file, with name of 'myfunc.m', called from the workspace or another function using myfunc().
4. All variables will be local to the function.

(written to a file in the working directory ,"mysqr.m")

```
function y = mysqr(x)
y=x*x;
```

ODE sample; sine and cosine

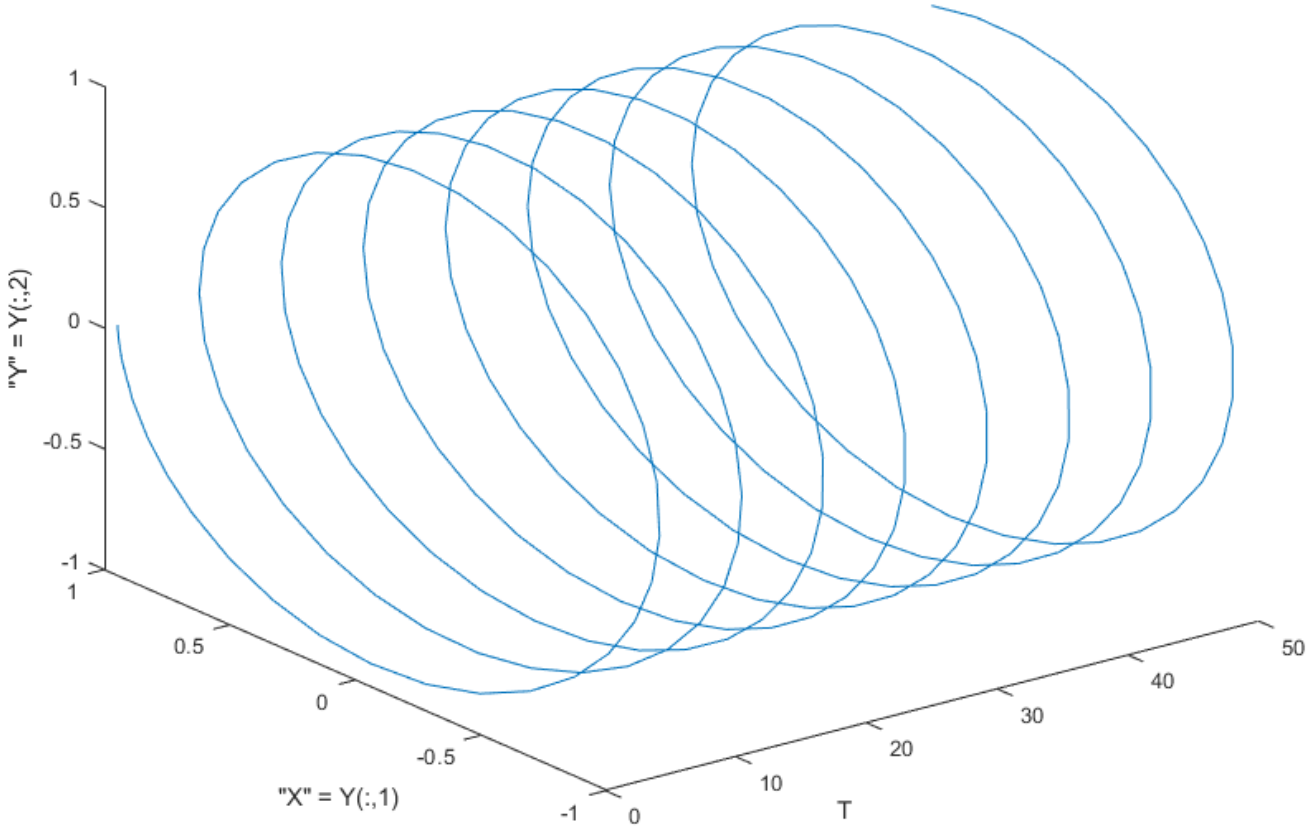
(file "spiralrun.m")

```
Tspan = [1 50]; %[start end]
Y0 = [1 0];  % initial conditions, 2 variables
[T,Y] = ode45(@spiral,Tspan, Y0);
```

(file "spiral.m")

```
function dy = spiral(t,y)
dy=zeros(2,1); % column vector w/ 2 rows
dy(1)=y(2);
dy(2)=-y(1);
```

ODE example - spiral




```
plot3(T,Y(:,1),Y(:,2));
```

Passing variables

*The **global** command can be used to access workspace variables not passed to the function*

(file "spiralrun.m")

```
global mag  %passed parameter
mag = .1;
Tspan = [1 50]; %[start end]
Y0 = [1 0];  % initial conditions, 2 variables
[T,Y] = ode45(@spiral,Tspan, Y0);
```

(file "spiral.m")

```
function dy = spiral(t,y)
global mag
dy=zeros(2,1); % column vector w/ 2 rows
dy(1)=mag*y(2);
dy(2)=-mag*y(1);
```

Numerical stiffness

$$\frac{d\sigma_1}{d\tau} = \nu - \frac{k_2 + k_{-1}}{k_2} u_1 \sigma_1 + \frac{k_{-1}}{k_2} u_2,$$

$$\frac{d\sigma_2}{d\tau} = \alpha \left[u_2 - \frac{k_{-3}}{k_2} \sigma_2^{\gamma} (1 - u_1 - u_2) + \frac{k_{-3}}{k_2} u_1 \right] - \eta \sigma_2,$$

$$0 = \epsilon \frac{du_1}{d\tau} = u_2 - \sigma_1 u_1 + \frac{k_{-3}}{k_2 + k_{-1}} \left[\sigma_2^{\gamma} (1 - u_1 - u_2) - u_1 \right],$$

$$0 = \epsilon \frac{du_2}{d\tau} = \sigma_1 u_1 - u_2,$$

- Disparity of timescales leads to problems in stepsize selection

Runge-Kutta 4,5 method

- Good, general purpose solver.
- Looks at dy/dt at certain points

$$y' = f(t, y), \quad y(t0) = y0.$$

$$y_{n+1} = y_n + (h/6)^* (k_1 + 2k_2 + 2k_3 + k_4); \; t_{n+1} = t_n + h$$

$$k_1 = f(t_n, y_n)$$

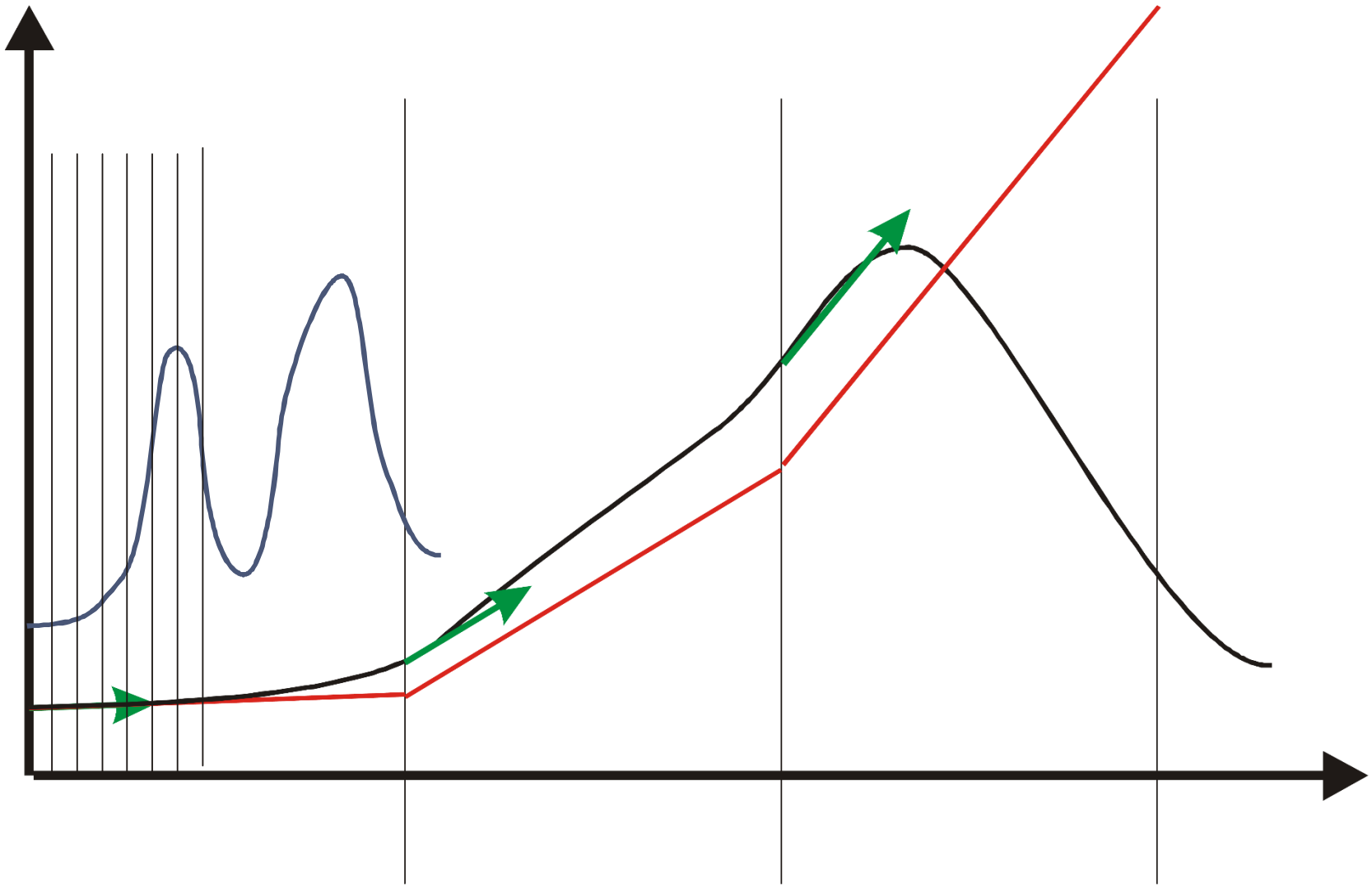
$$k_2 = f(t_n + (h/2), y_n + (h/2)^* k_1)$$

$$k_3 = f(t_n + (h/2), y_n + (h/2)^* k_2)$$

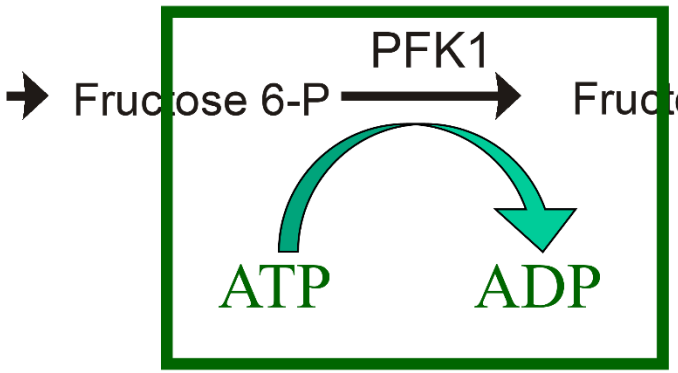
$$k_4 = f(t_n + h, y_n + h^* k_3).$$

- “explicit solution”. “Implicit” variations are around
- selection of timestep is critical, balancing rounding and interpolation/extrapolation errors

Numerical stiffness



Your goal



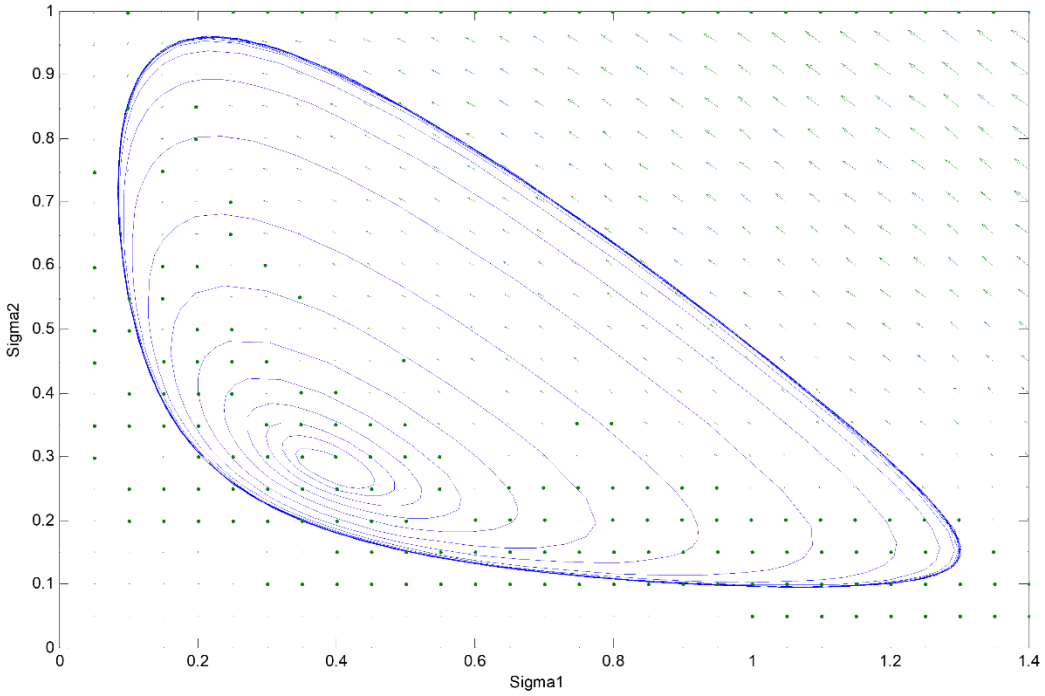
$E = PFK1$

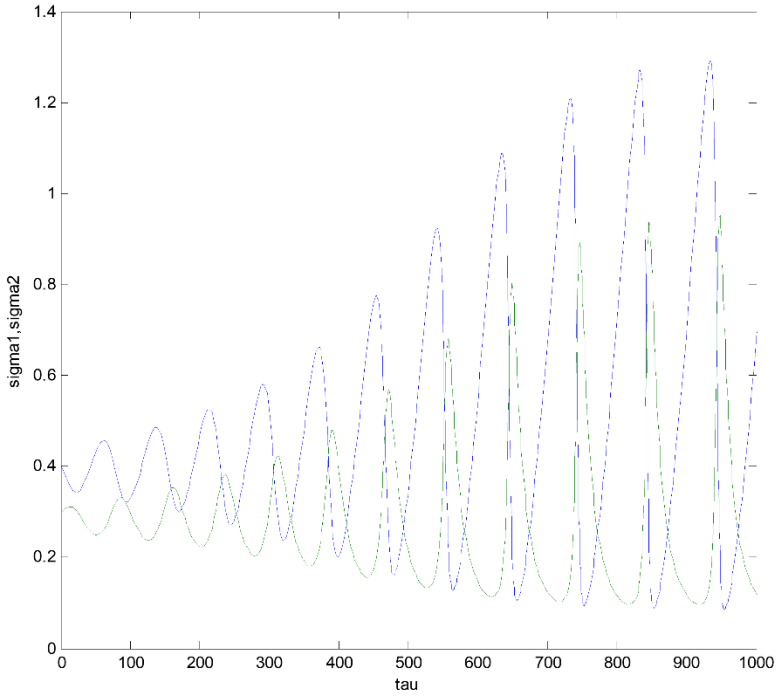
$S_1 = ATP$

$S_2 = ADP$

$x_1 = ES_2^{\gamma}$

$x_2 = S_1 ES_2^{\gamma}$





Your goal: the system

$$\frac{d\sigma_1}{d\tau} = \nu - f(\sigma_1, \sigma_2),$$

$$\frac{d\sigma_2}{d\tau} = \alpha f(\sigma_1, \sigma_2) - \eta\sigma_2.$$

$$\frac{\sigma_1 \sigma_2^{\gamma}}{\sigma_2^{\gamma}\sigma_1 + \sigma_2^{\gamma} + 1} = f(\sigma_1, \sigma_2)$$

Starting point

(file "spiralrun.m")

```
global mag  %passed parameter
mag = .1;
Tspan = [1 200]; %[start end]
Y0 = [1 2];  % initial conditions, 2 variables
[T,Y] = ode45(@spiral,Tspan, Y0);
```

(file "spiral.m")

```
function dy = spiral(t,y)
global mag
dy=zeros(2,1); % column vector w/ 2 rows
dy(1)=mag*y(2);
dy(2)=-mag*y(1);
```