BENDUKIDZE CAMPUS

18.11.2024 | 18:00

SCIENTIA • LABOR • LIBERTAS

თავისუფალი უნივერსიტეტი

SCIENTIA • LABOR • LIBERTAS

აგრარული უნივერსიტეტი

Computer Modelling of Physical systems

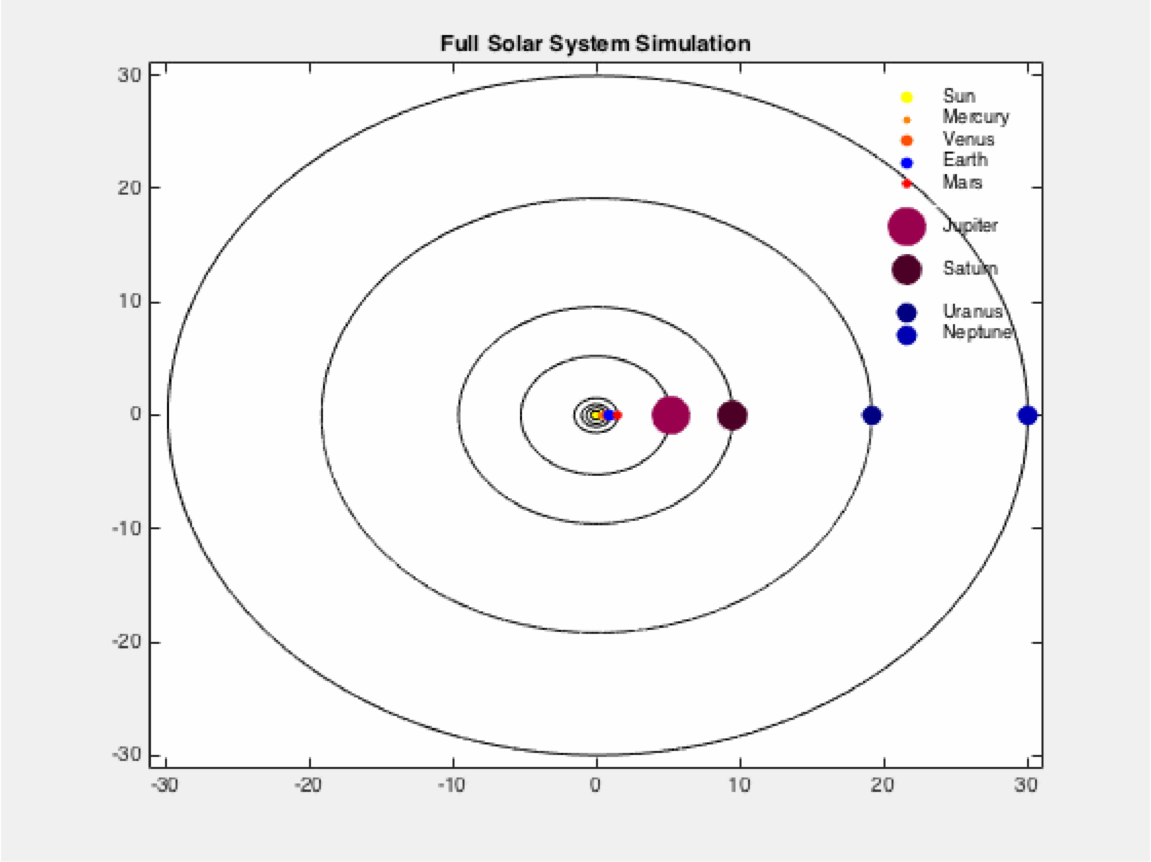
Computer Modelling of Physical systems

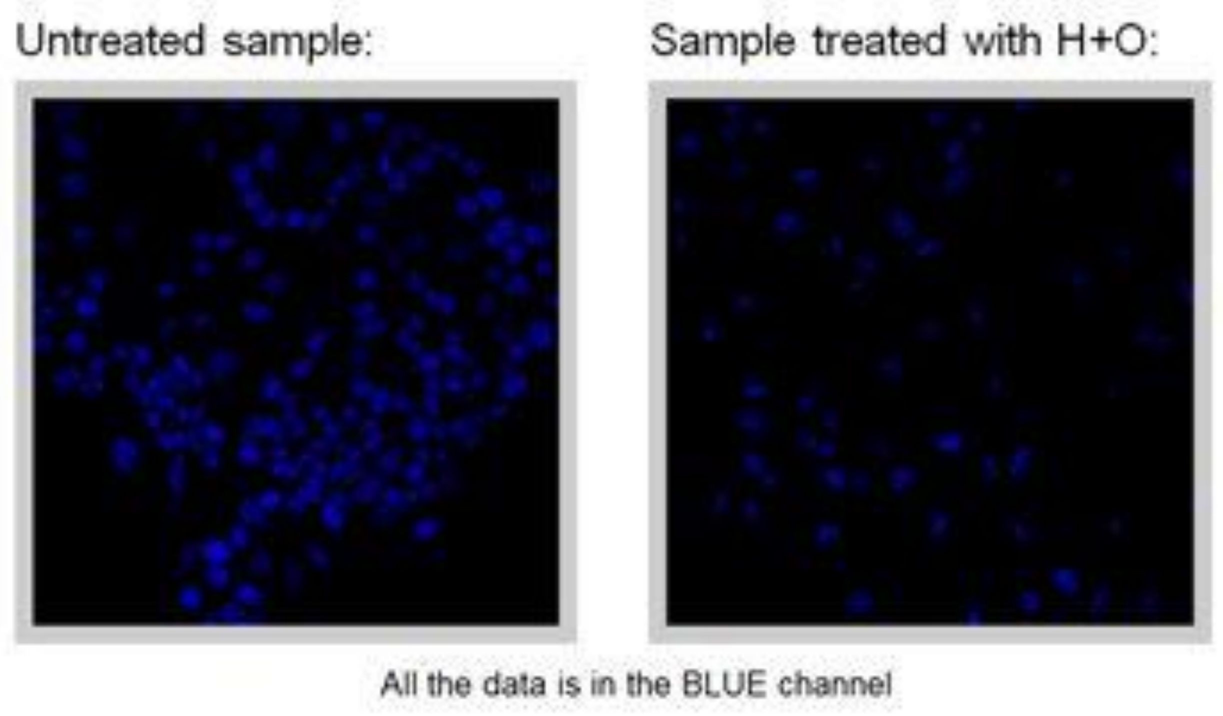
Seminar Series vol.1

Read and prepared by Zurabi Ugulava

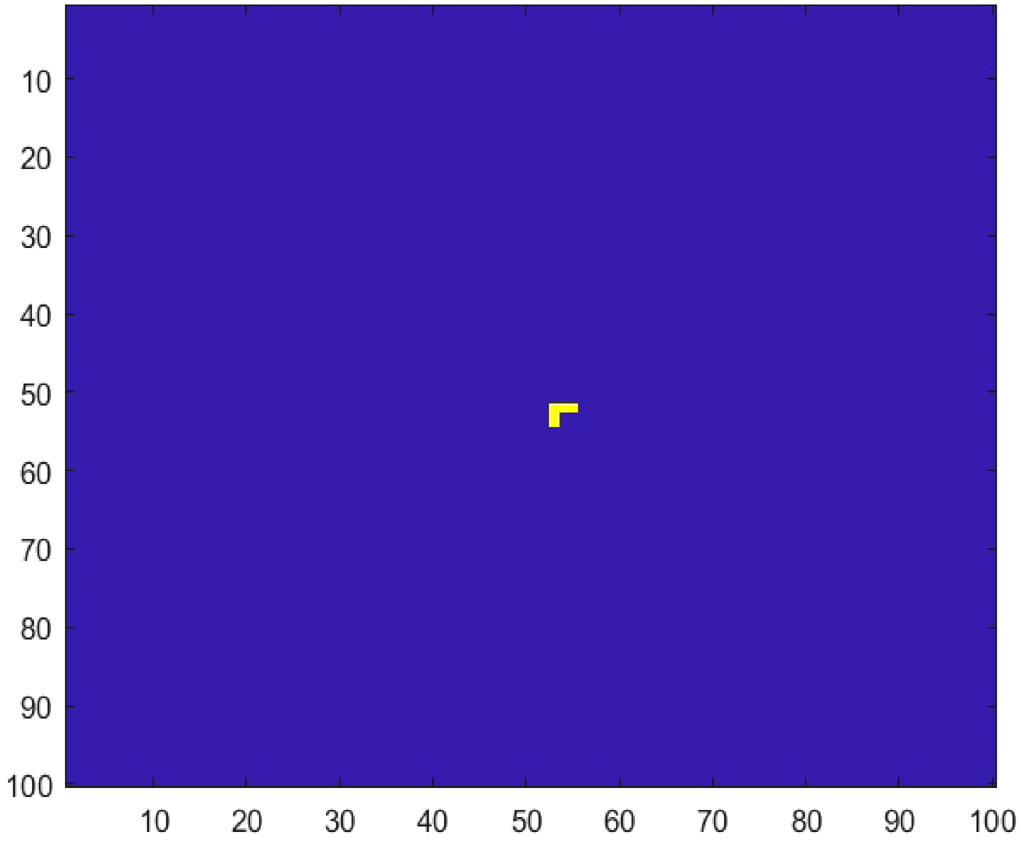
What to expect next?

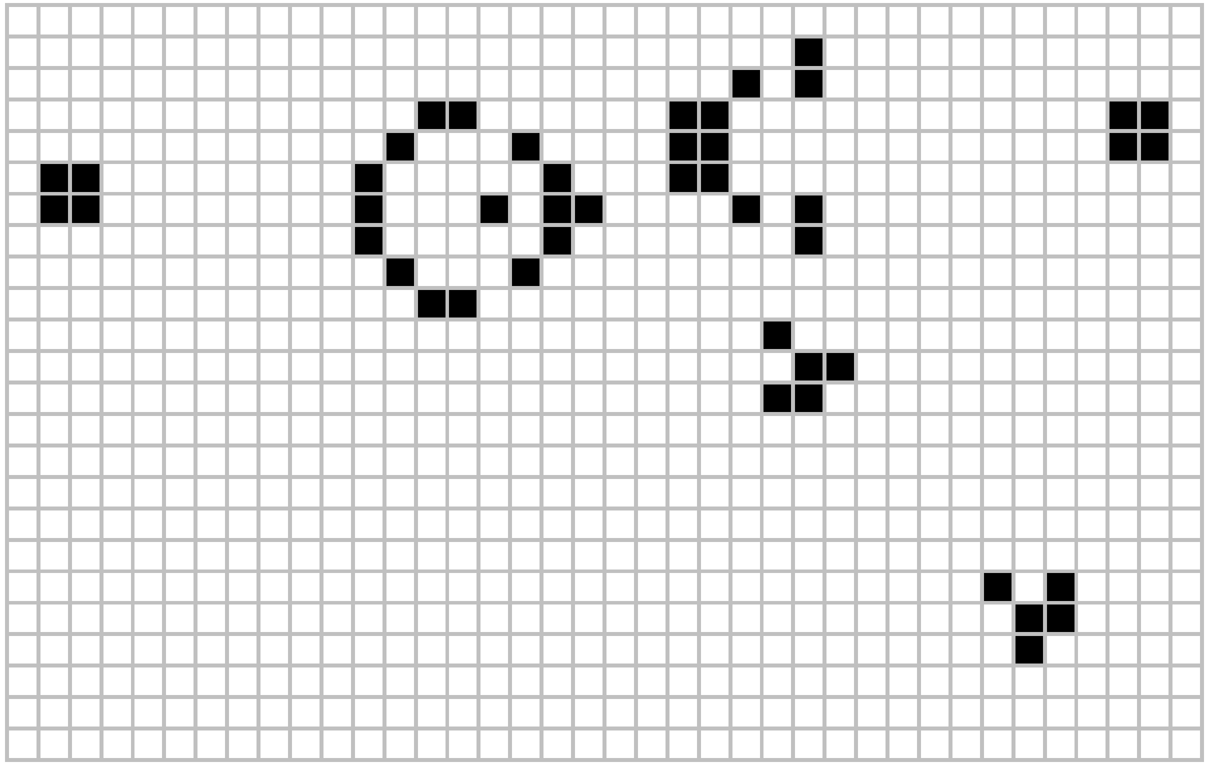
- Three Body problem/ Astrophysical insight
- Introduction to chaotic systems
- From Conwoy game to Ising 2D model MCS
- Data analysis module
- And many more…



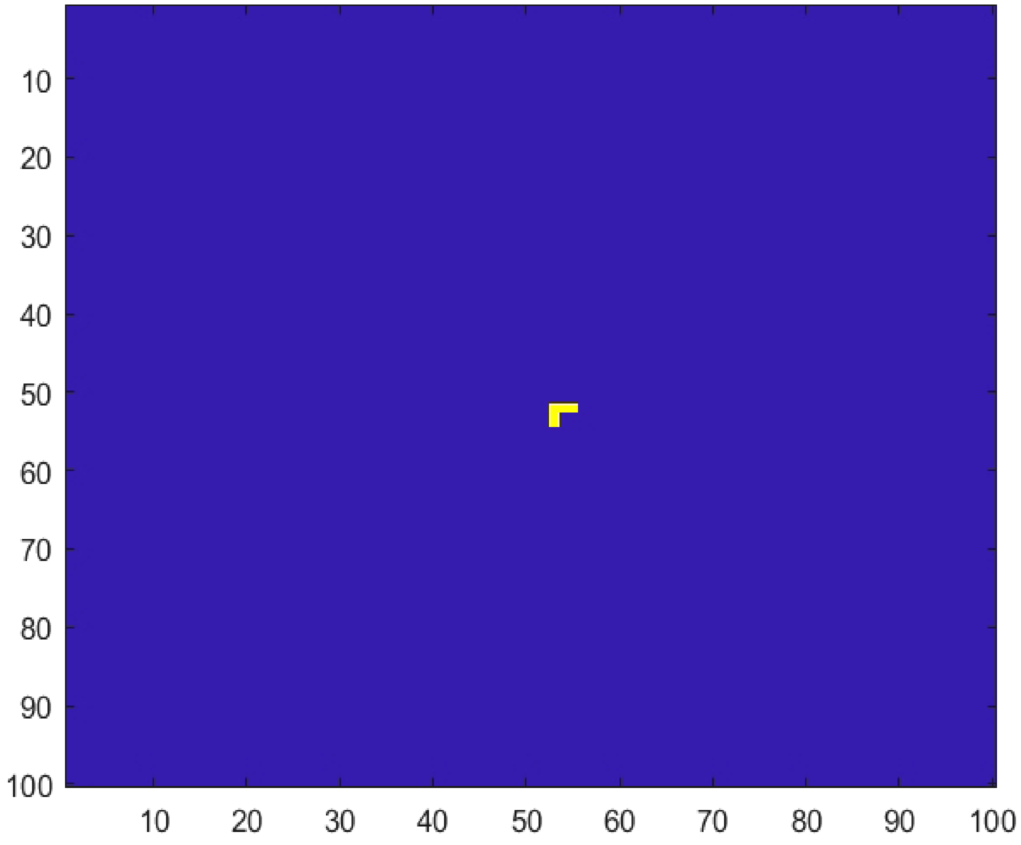


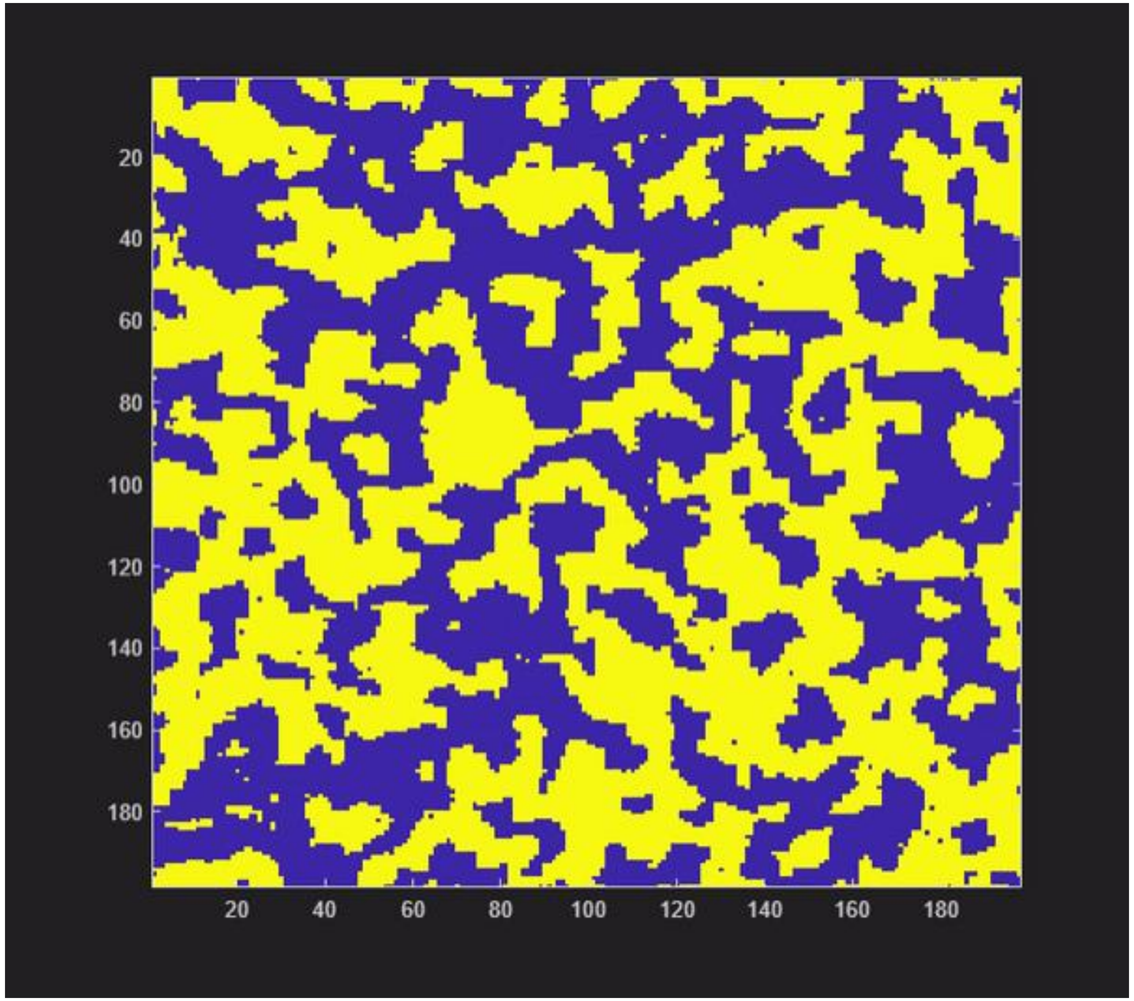
https://www.mathworks.com/company/technical-articles/using-image-processing-and-statistical-analysis-to-quantify-cell-scattering-for-cancer-drug-research.html

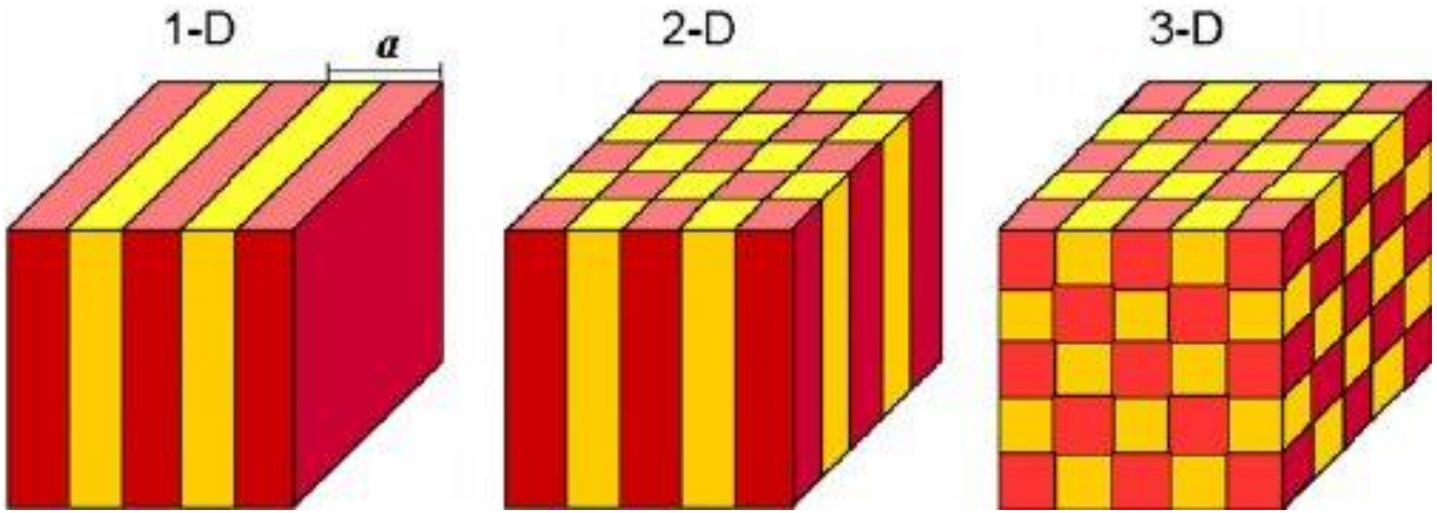




Bill Gosper's Glider Gun in action—a variation of Conway’s Game of Life. This image was made by using Life32 v2.15 beta, by Johan G. Bontes.

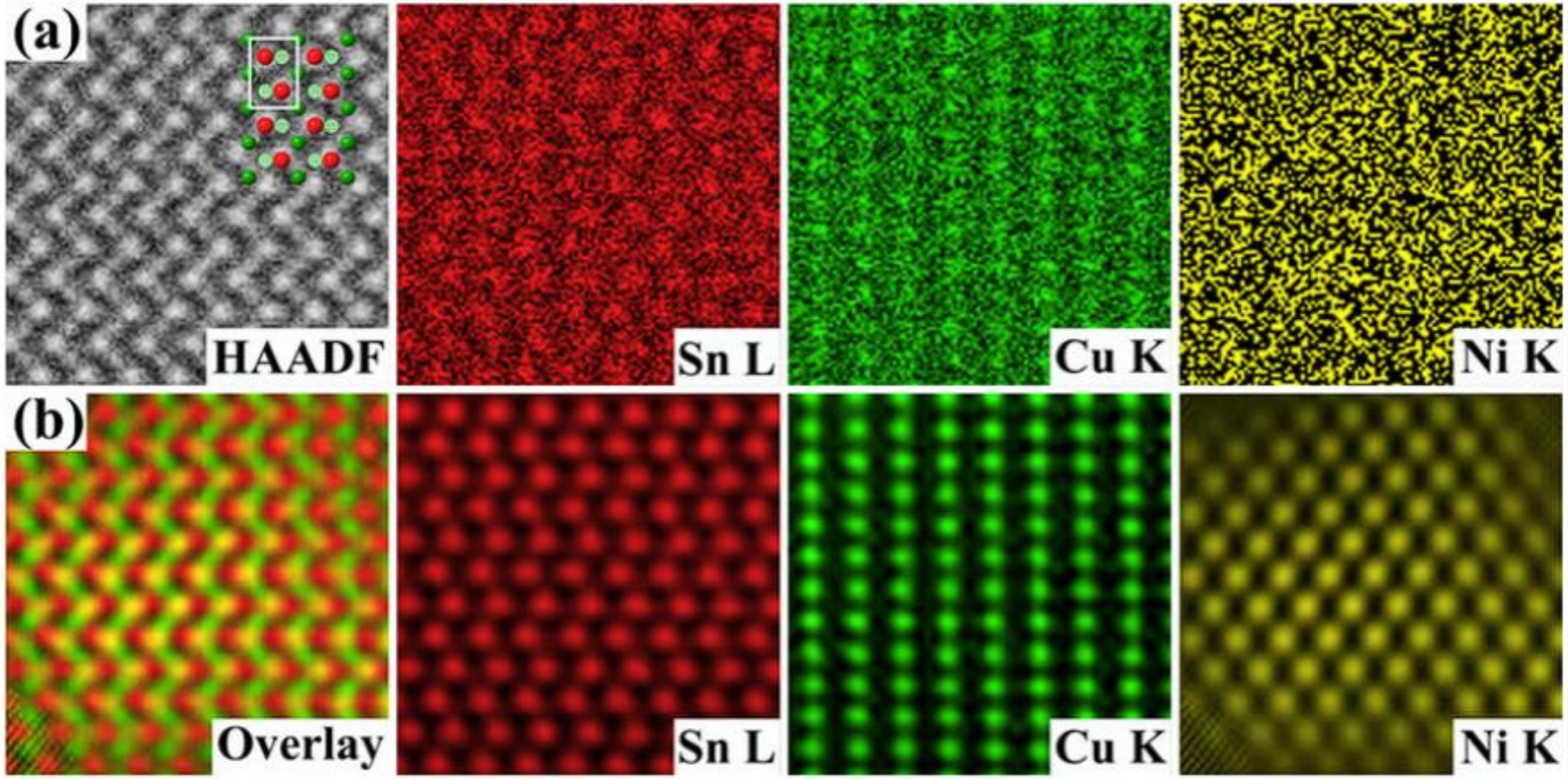

10
20
30
40
50
60
70
80
90
100
10 20 30 40 50 60 70 80 90 100


20
40
60
80
100
120
140
160
180
20 40 60 80 100 120 140 160 180



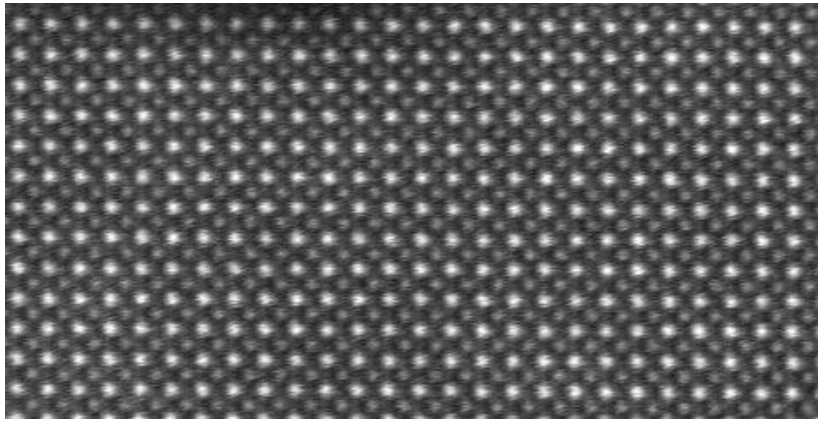
Intro to molecular dynamics

Physics of planar lattice

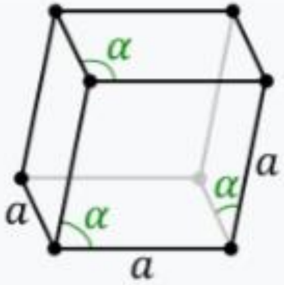
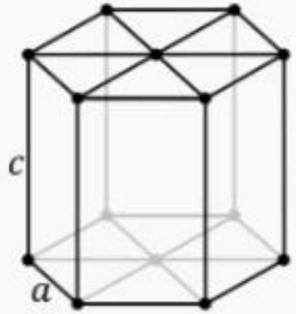


(a) The original atomic-resolution image and elemental mapping: Sn (red), Cu (green), Ni (yellow); (b) Noise reduction image of atomic-resolution elemental mapping: Overlay, Sn (red), Cu (green), Ni (yellow).

Matsumura, Syo. (2019). Atom locations in a Ni doped η-$(Cu,Ni)_6Sn_5$ intermetallic compound. Scripta Materialia. 158. 1-5. 10.1016/j.scriptamat.2018.08.020.



Strontium Titanate

Crystal system	Trigonal		Hexagonal
Lattice system	 Rhombohedral	 Hexagonal	
Example	 Dolomite (white)	 α-Quartz	 Beryl

Physical description

$$M\frac{dv}{dt} = \sum F_i$$

$$E = \frac{1}{2}mv^2 + \frac{1}{2}kx^2$$

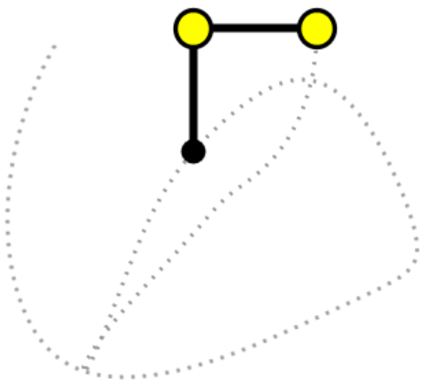
$$p = \sum m_i v_i$$

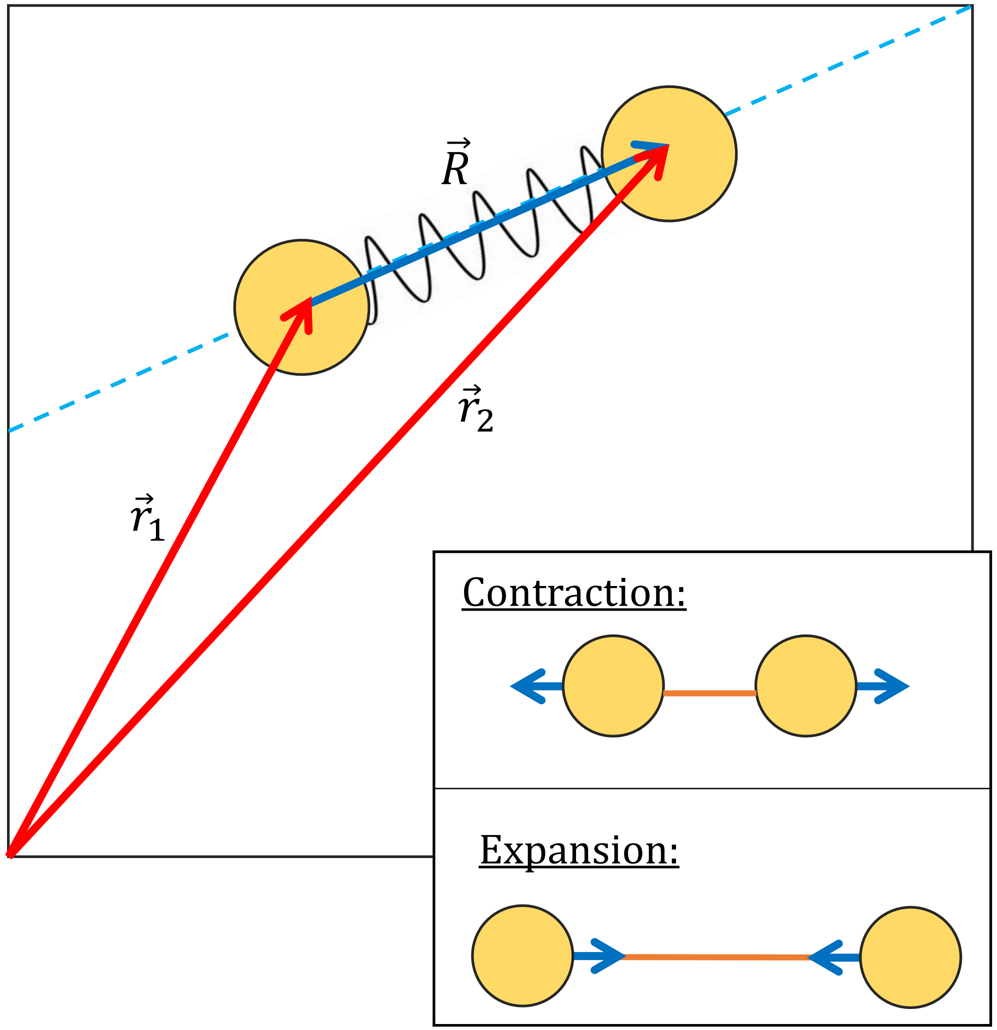
Translate physics into code

springmasssystem.m

```
obj=VideoWriter('animation.avi');
obj.Quality=100;
obj.FrameRate=60;
open(obj);

for i=1:N

    %Acceleration increment
    if abs(y(i)-x(i))>l0
        x2(i)=k/m*abs(y(i)-x(i)-l0);
        y2(i)=-k/M*abs(y(i)-x(i)-l0);
    else
        x2(i)=-k/m*abs(y(i)-x(i)-l0);
        y2(i)=k/M*abs(y(i)-x(i)-l0);
```

Extract data in the desired form





Establishing the system in space:

$$\vec{r}_1 = (x_1, y_1, z_1)$$

$$\vec{r}_2 = (x_2, y_2, z_2)$$

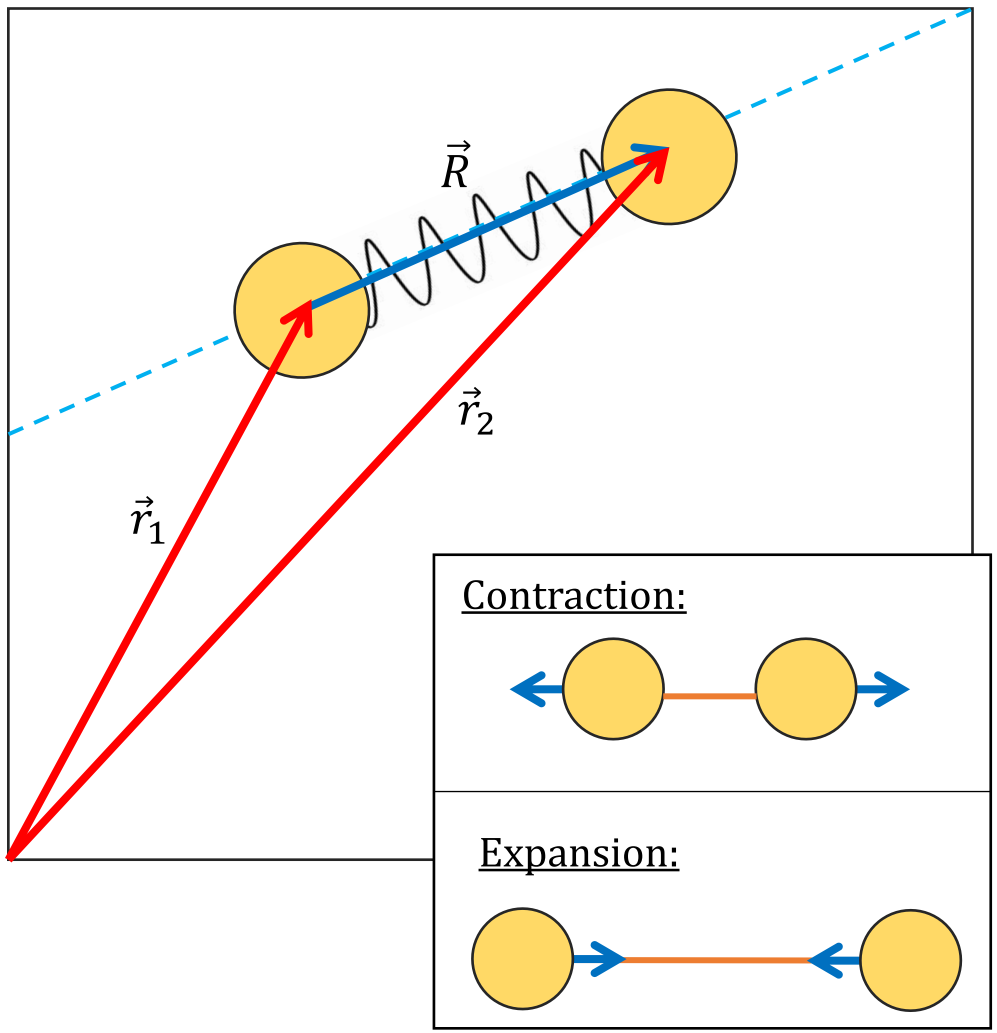
$$\vec{R} \equiv \vec{r}_{12} = \vec{r}_2 - \vec{r}_1$$

Ordinary Hook's Law:

$$x = |\vec{R}| - L_0$$

Force ~ displacement

$$\vec{F} = k\vec{x} = k\left(1 - \frac{|\vec{R}|}{L_0}\right)\hat{R}$$



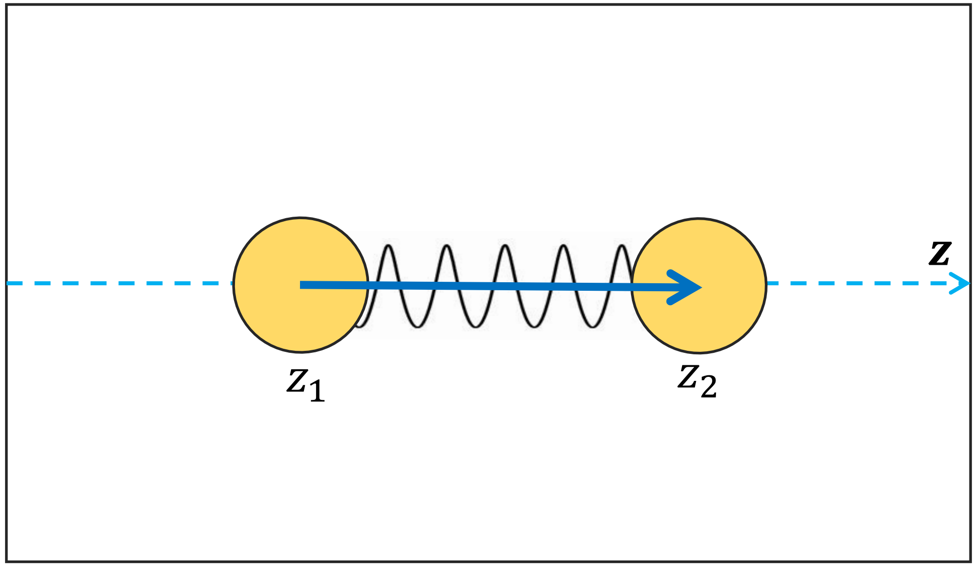
Ordinary Hook's Law:

$x = \lvert\vec{R}\rvert - L_0$	Force ~ displacement
$\vec{F} = k\vec{x} = kL_0\left(1 - \frac{\lvert\vec{R}\rvert}{L_0}\right)\hat{R}$	

Equations of motion:

$$m_1 \frac{d^2\vec{r}_1}{dt^2} = \vec{F} = kL_0\left(1 - \frac{|\vec{R}|}{L_0}\right)\hat{R}$$

$$m_2 \frac{d^2\vec{r}_2}{dt^2} = -\vec{F} = -kL_0\left(1 - \frac{|\vec{R}|}{L_0}\right)\hat{R}$$



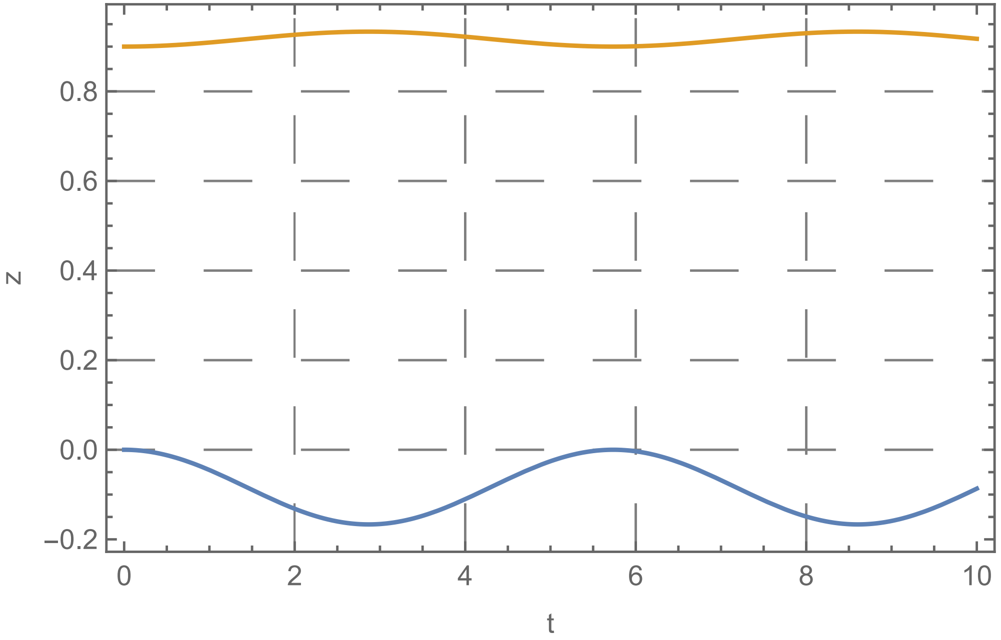
Equations of motion:

$$
\begin{cases}
m_1 \dfrac{d^2\vec{z}_1}{dt^2} = k(z_2 - z_1 - L_0)\hat{R} \\
m_2 \dfrac{d^2\vec{z}_2}{dt^2} = -k(z_2 - z_1 - L_0)\hat{R}
\end{cases}
$$

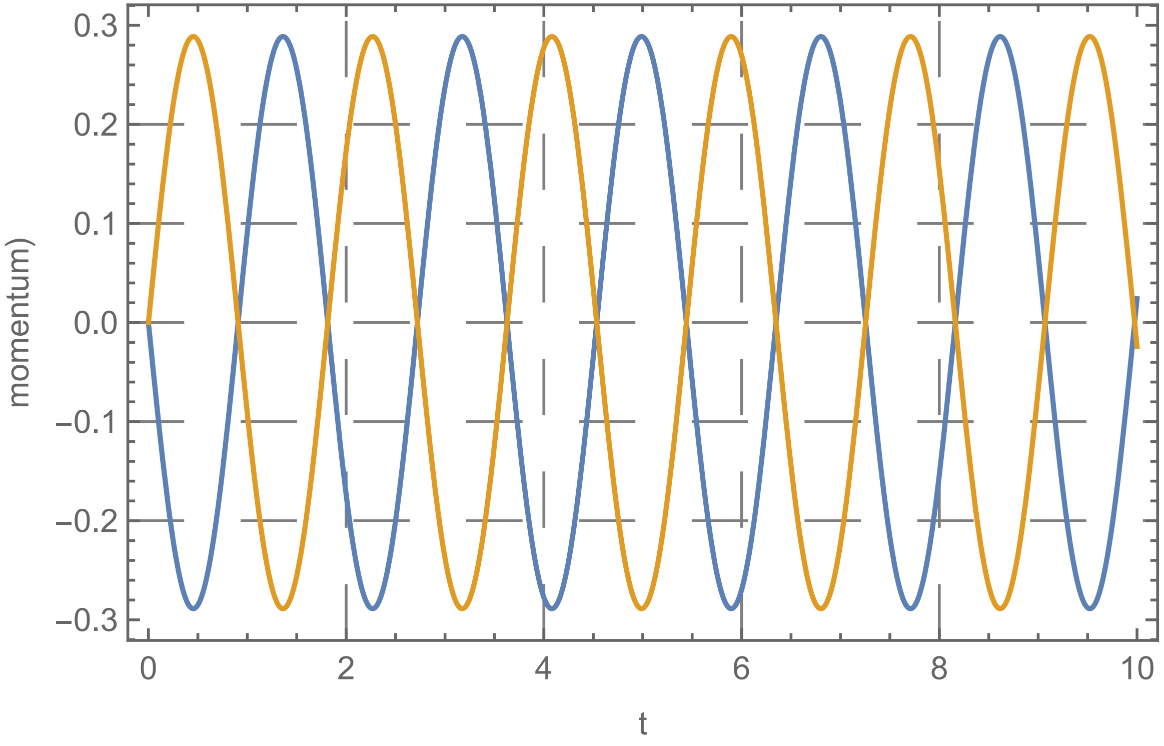
$$z_1(t) = A_1 \sin(\omega t + \varphi_1)$$

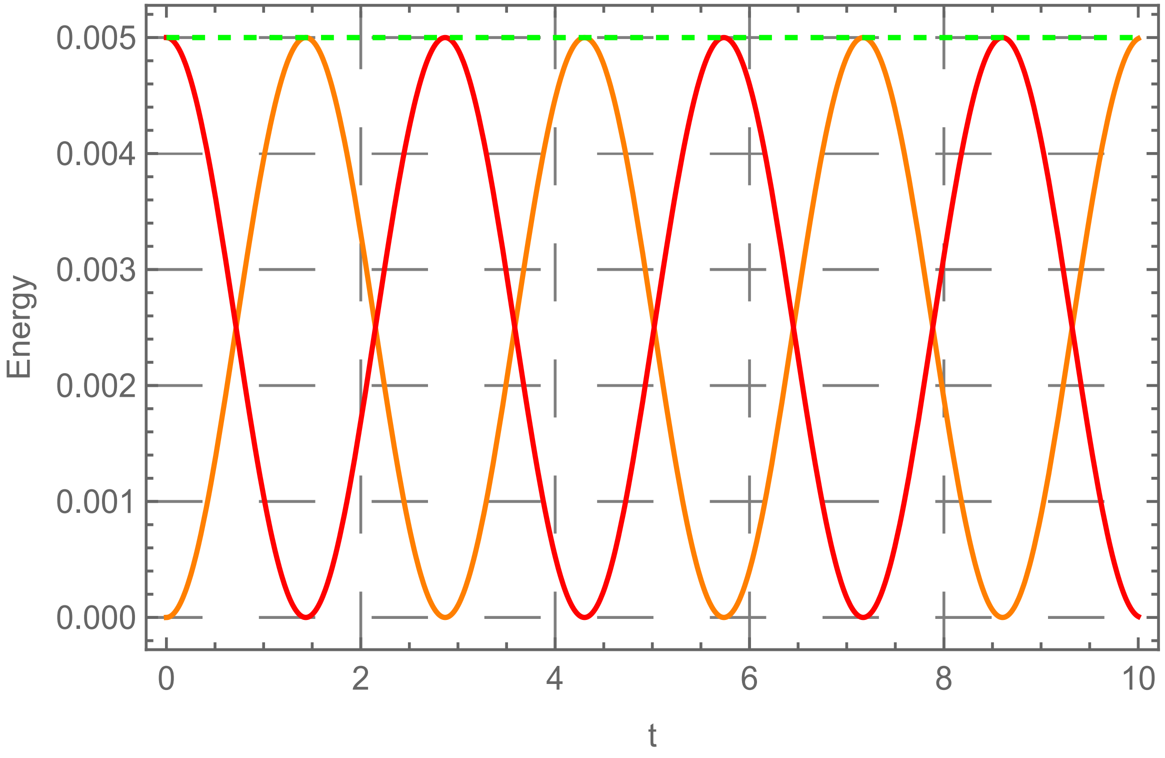
$$z_2(t) = A_2 \sin(\omega t + \varphi_2)$$

$$\omega^2 = \frac{\mu}{k} = \frac{1}{k} \cdot \frac{m_1 m_2}{m_1 + m_2}$$

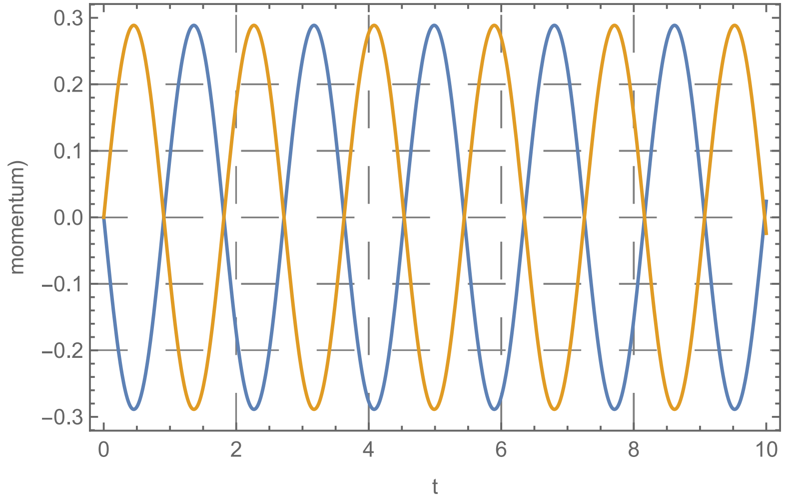


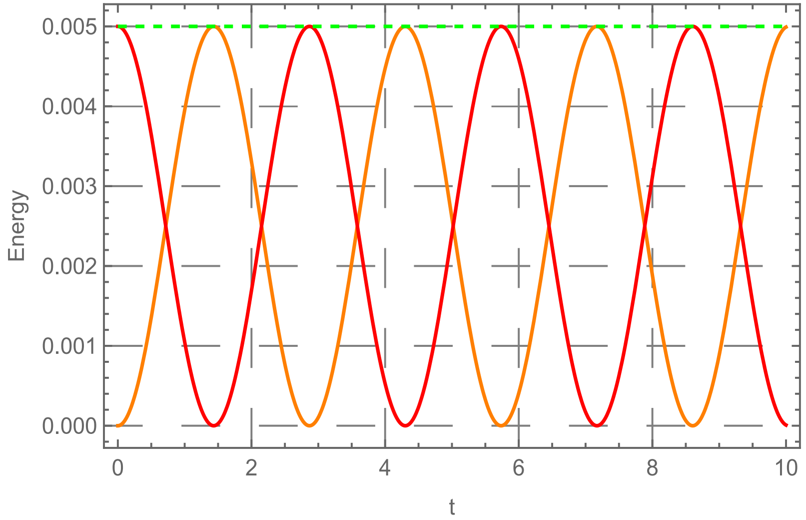
What to observe in periodic system:

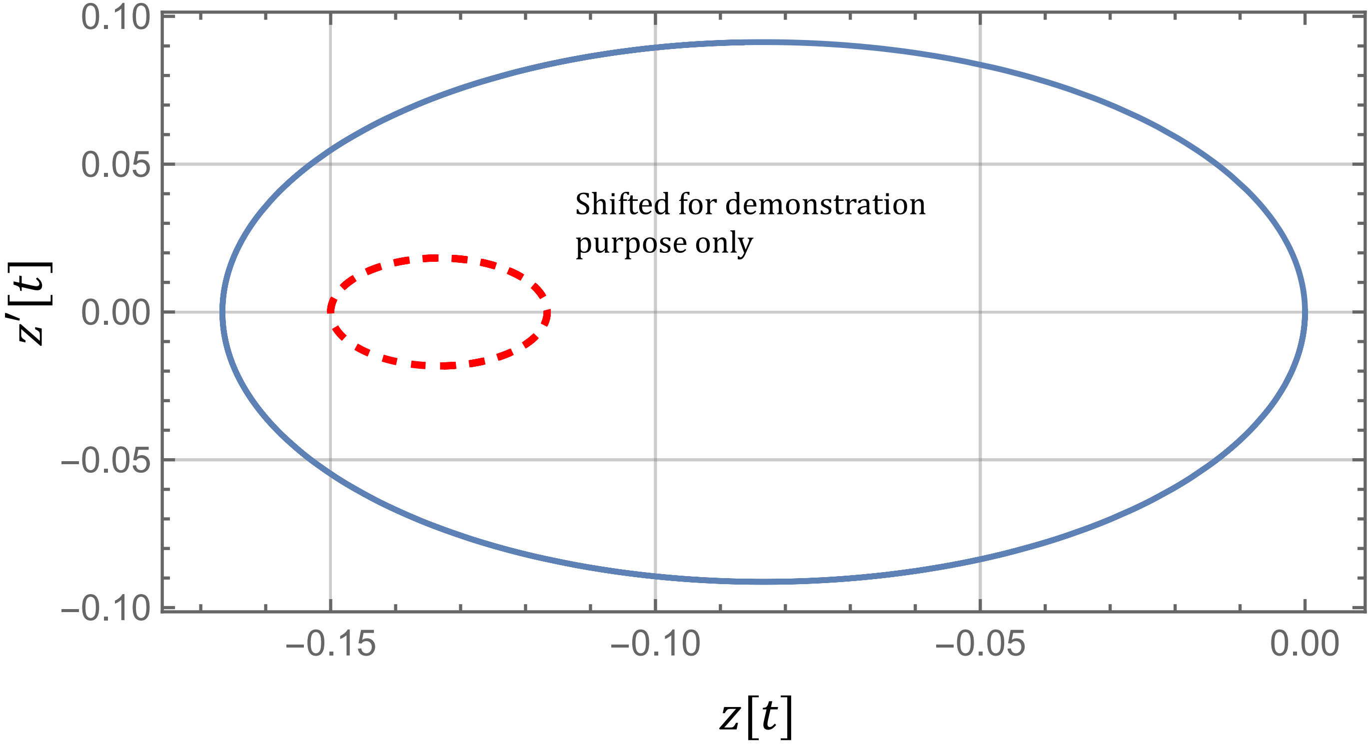




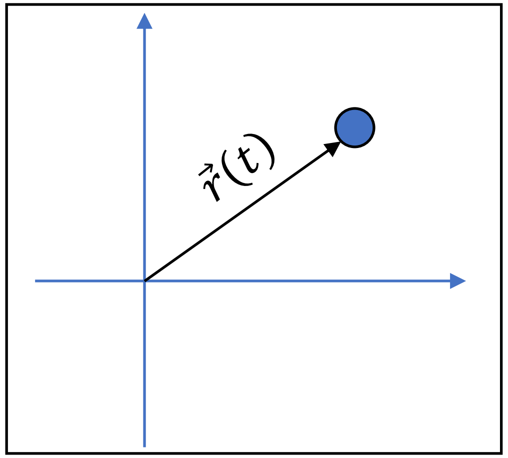
What to observe in periodic system:







Physical description:



Governing physics described by Differential equations!

But how do we handle differential equations?

Manual Integration/

$$f'(x) - \frac{1}{x} = 0 \qquad f(x) = \ln(x) + \mathcal{C}$$

$$f''(x) - x = 0 \qquad f(x) = \frac{1}{6}x^3 + \frac{1}{2}\mathcal{C}_1 x^2 + \mathcal{C}_2$$

Another way to handle them ?

Let's remember Taylor series:

$$f(\vec{r}, t + \delta t) = f(\vec{r}, t) + \frac{1}{1!}\left(\frac{df}{dt}\right)\delta t + \frac{1}{2!}\left(\frac{d^2 f}{dt^2}\right)\delta t^2 + \cdots$$

$$\underset{t_0}{\boxed{f(\vec{r}_t, t)}} \to \underset{t_1}{\boxed{f(\vec{r}_{t+\delta t}, t + \delta t)}} \to \underset{t_2}{\boxed{f\left(\vec{r}_{t_1+\delta t}, t_1 + \delta t\right)}} \to \cdots$$

Let's remember Taylor series:

$$\boxed{f(\vec{r}, t+\delta t) = f(\vec{r}, t) + \frac{1}{1!}\left(\frac{df}{dt}\right)\delta t} + \frac{1}{2!}\left(\frac{d^2 f}{dt^2}\right)\delta t^2 + \cdots$$

1^{st} order approximation

Euler Algorithm:	RK-4 Algorithm:
$f(\vec{r}, t+\delta t) = f(\vec{r}, t) + f'(\vec{r}, t)\,\delta t$	$f(\vec{r}, t+\delta t) = f(\vec{r}, t) + \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4)\,\delta t$
$f'(\vec{r}, t+\delta t) = f'(\vec{r}, t) + f''(\vec{r}, t)\,\delta t$	$k_1 = f'(\vec{r}, t)$
○ ○ ○	$k_2 = f'\left(\vec{r} + \frac{1}{2}k_1\delta t, t + \frac{1}{2}\delta t\right)$
$f^{n-2}(\vec{r}, t+\delta t) = f^{n-2}(\vec{r}, t) + f^{n-1}(\vec{r}, t)\,\delta t$	$k_3 = f'\left(\vec{r} + \frac{1}{2}k_2\delta t, t + \frac{1}{2}\delta t\right)$
$f^{n-1}(\vec{r}, t+\delta t) = f^{n-1}(\vec{r}, t) + f^{n}(\vec{r}, t)\,\delta t$	$k_4 = f'(\vec{r} + k_3\delta t, t + \delta t)$

What is required to run a Simulation?

Main physical quantities

```
m1 = 10; m2 = 1; k = 10; l = 1;
```

```
le = Sqrt[(x2[t] - x1[t])^2 + (y2[t] - y1[t])^2 + (z2[t] - z1[t])^2];
```

Governing Equations

```
eqs = {
   m1 * x1''[t] - k * (1 - l / le) * (x2[t] - x1[t]) == 0, m1 * y1''[t] - k * (1 - l / le) * (y2[t] - y1[t]) == 0,
   m1 * z1''[t] - k * (1 - l / le) * (z2[t] - z1[t]) == 0, m2 * y2''[t] + k * (1 - l / le) * (x2[t] - x1[t]) == 0,
   m2 * x2''[t] + k * (1 - l / le) * (y2[t] - y1[t]) == 0, m2 * z2''[t] + k * (1 - l / le) * (z2[t] - z1[t]) == 0};
```

Initial Conditions

```
init = {
   x1[0] == 2, y1[0] == 0, z1[0] == 0,
   x2[0] == 1.1, y2[0] == 0, z2[0] == 0,
   x1'[0] == 0, y1'[0] == 0, z1'[0] == 0,
   x2'[0] == 0, y2'[0] == 0, z2'[0] == 0};
```

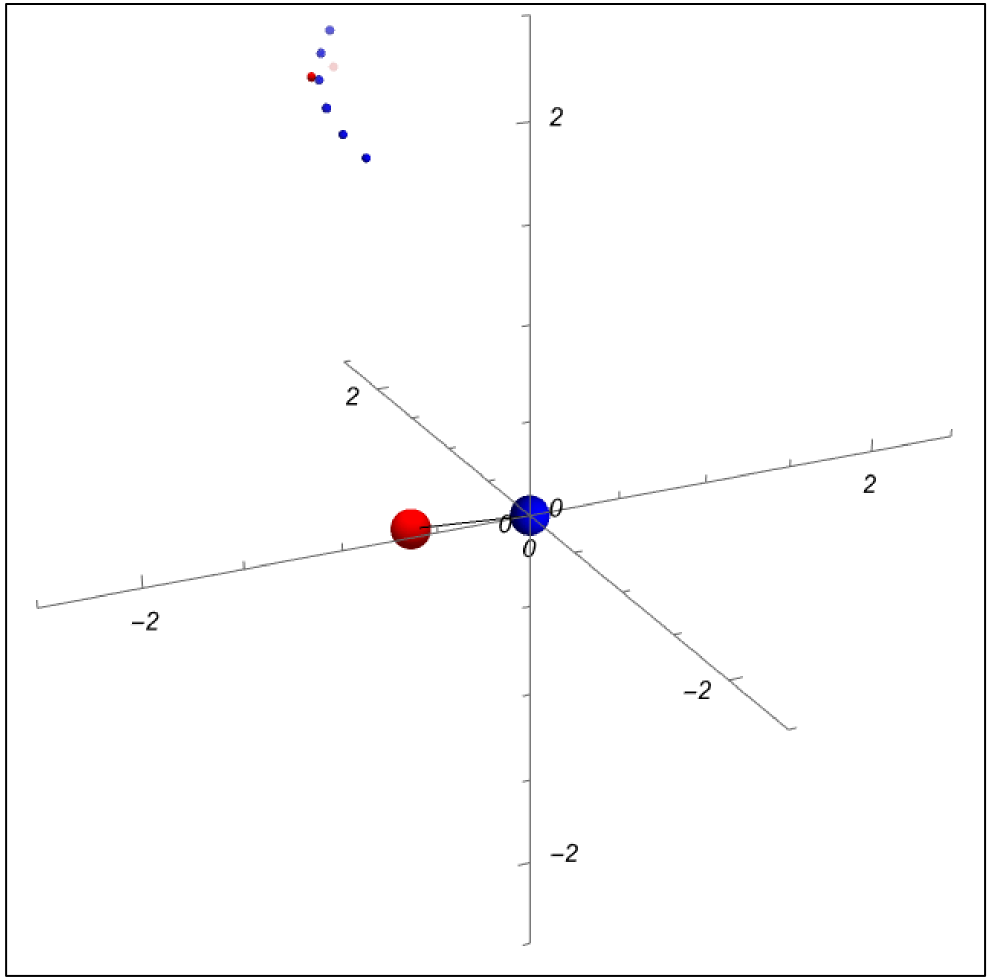
Solver Algorithm

```
sol = NDSolve[Join[eqs, init], {x1, y1, z1, x2, y2, z2}, {t, 0, 100}];
```

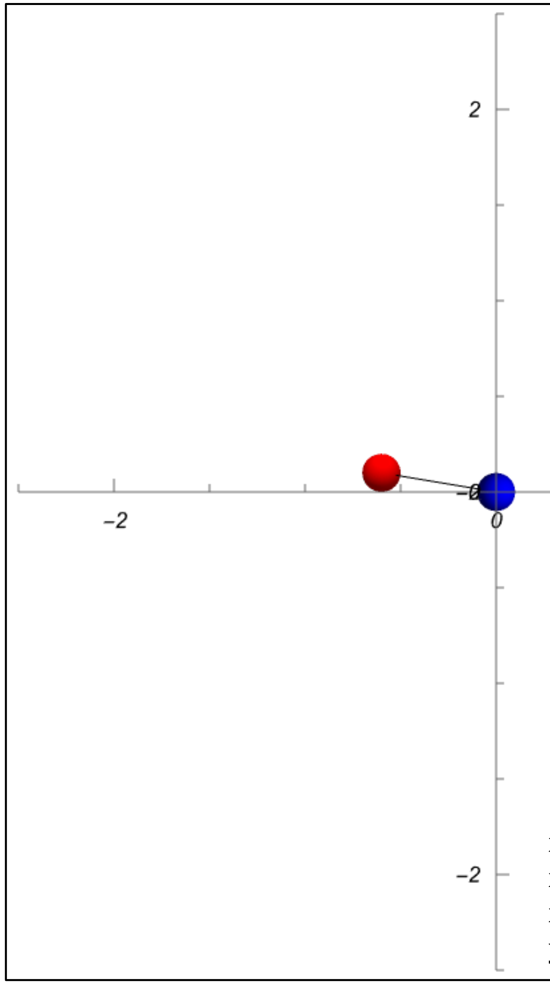
Visualization

```
Plot[Evaluate[{x1[t]}] /. sol, {t, 0, 10}]
```

Trajectory in 3D Space:

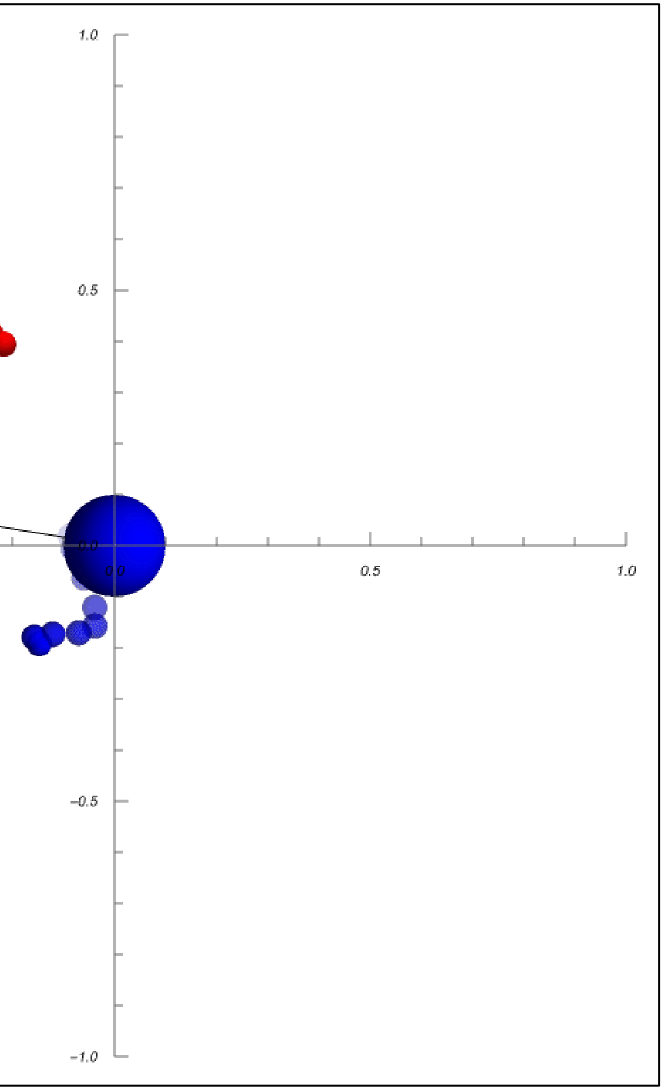


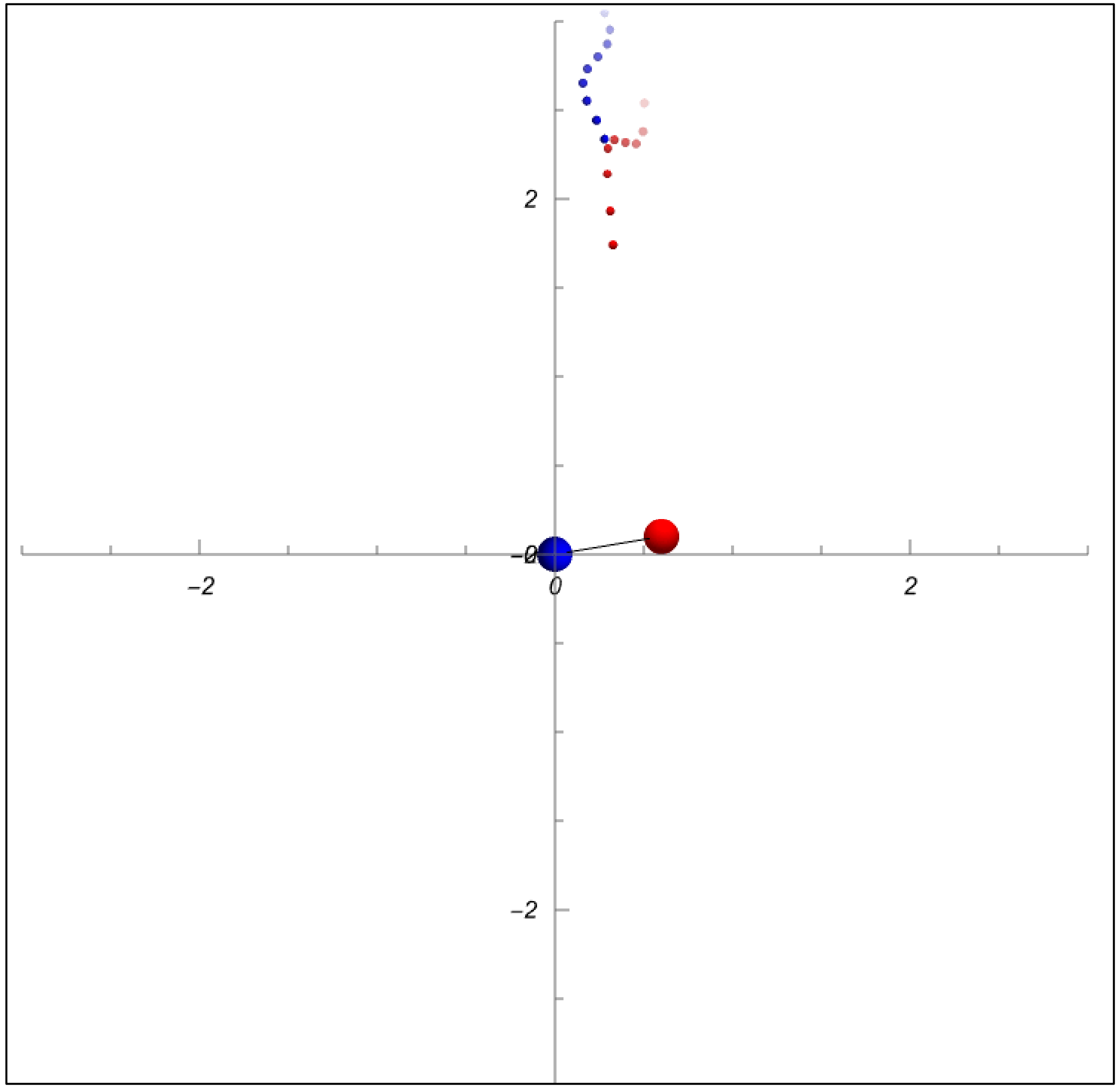
2D Projection



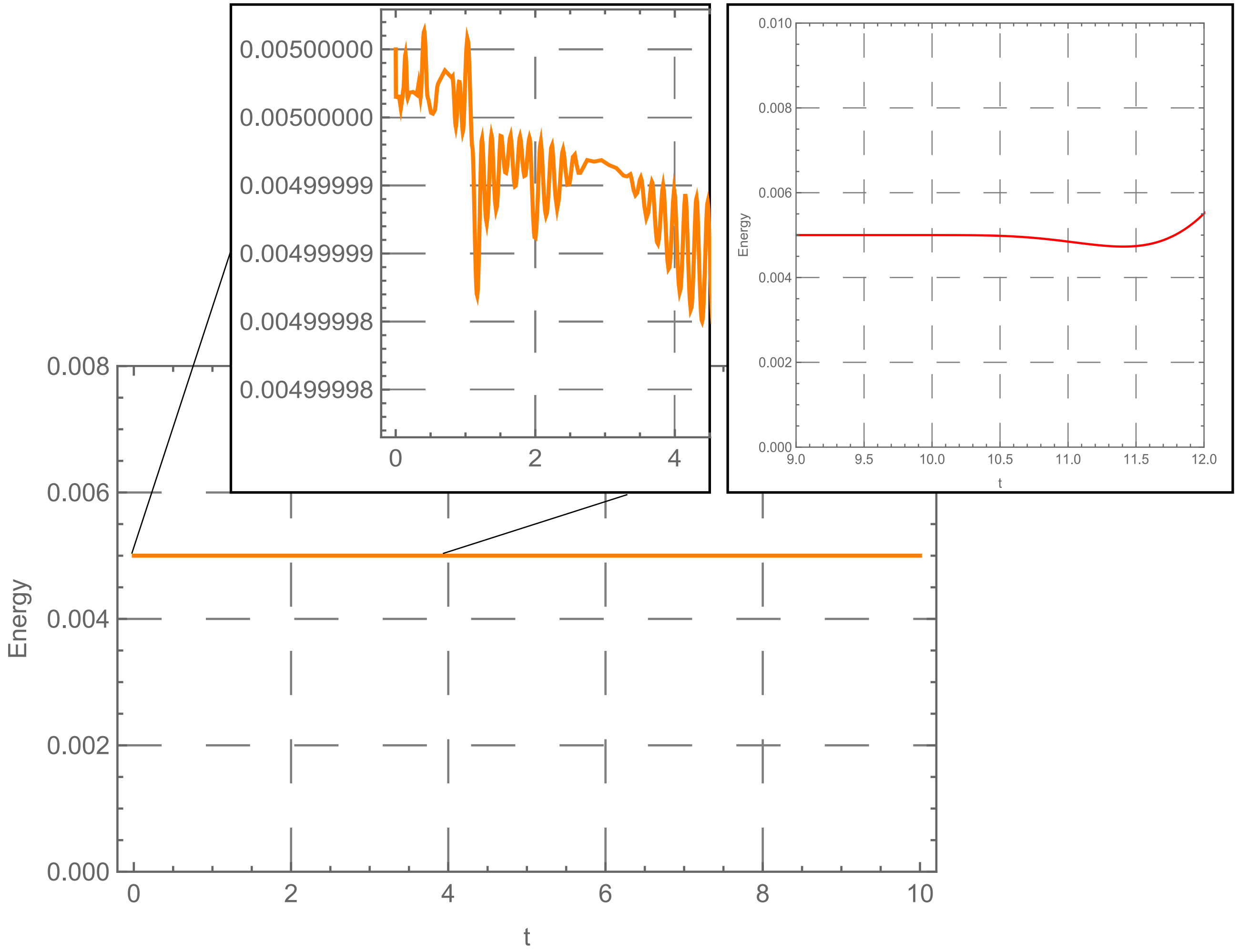
x1[0] = -0.6, y1[0] = 0.1,
x2[0] = y2[0] = z1[0]= z2[0] = 0
x1'[0] = x2'[0]= z2'[0] = 0
y1'[0] = y2'[0]= 0.1, z1'[0] = 0.2

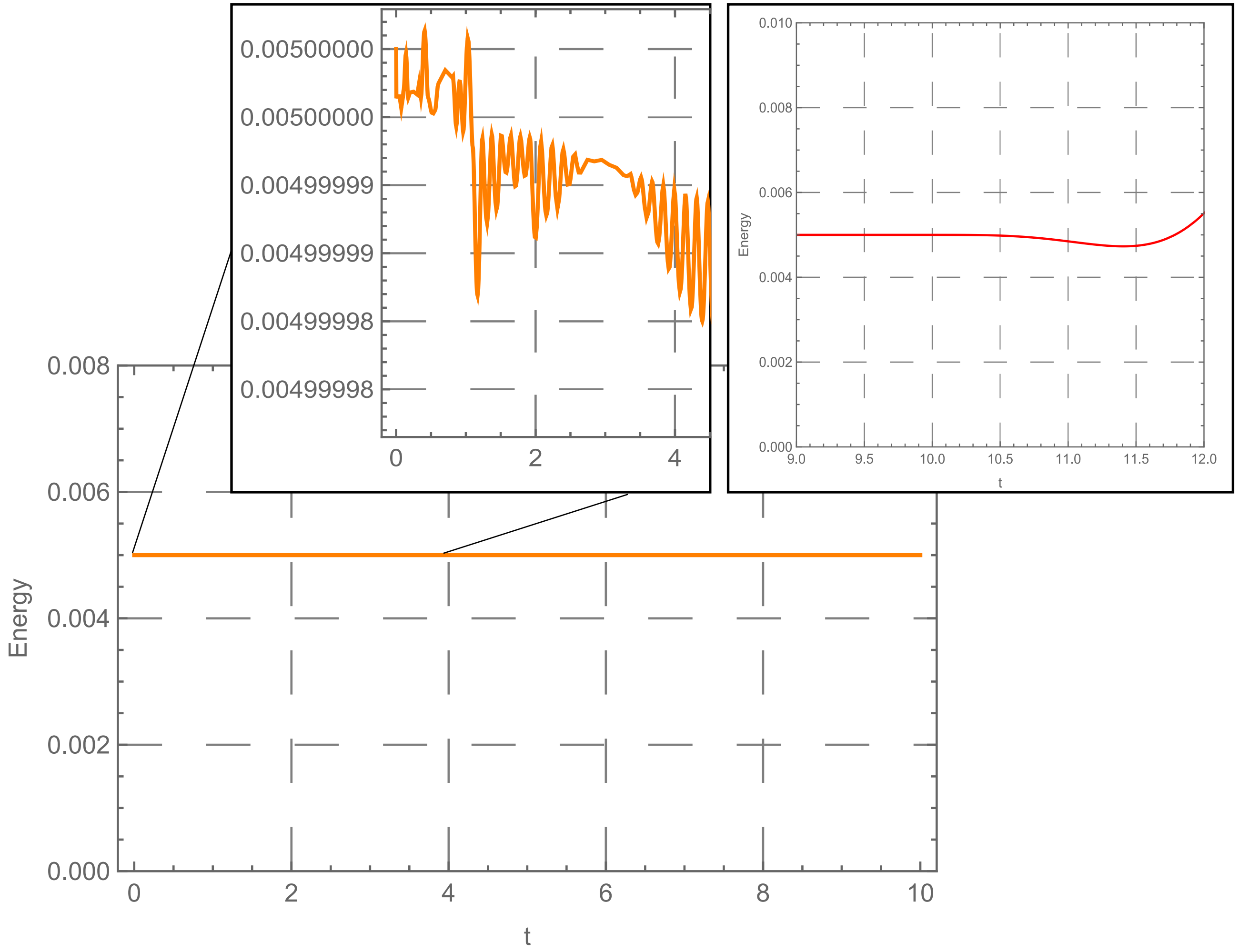
Parametric plots

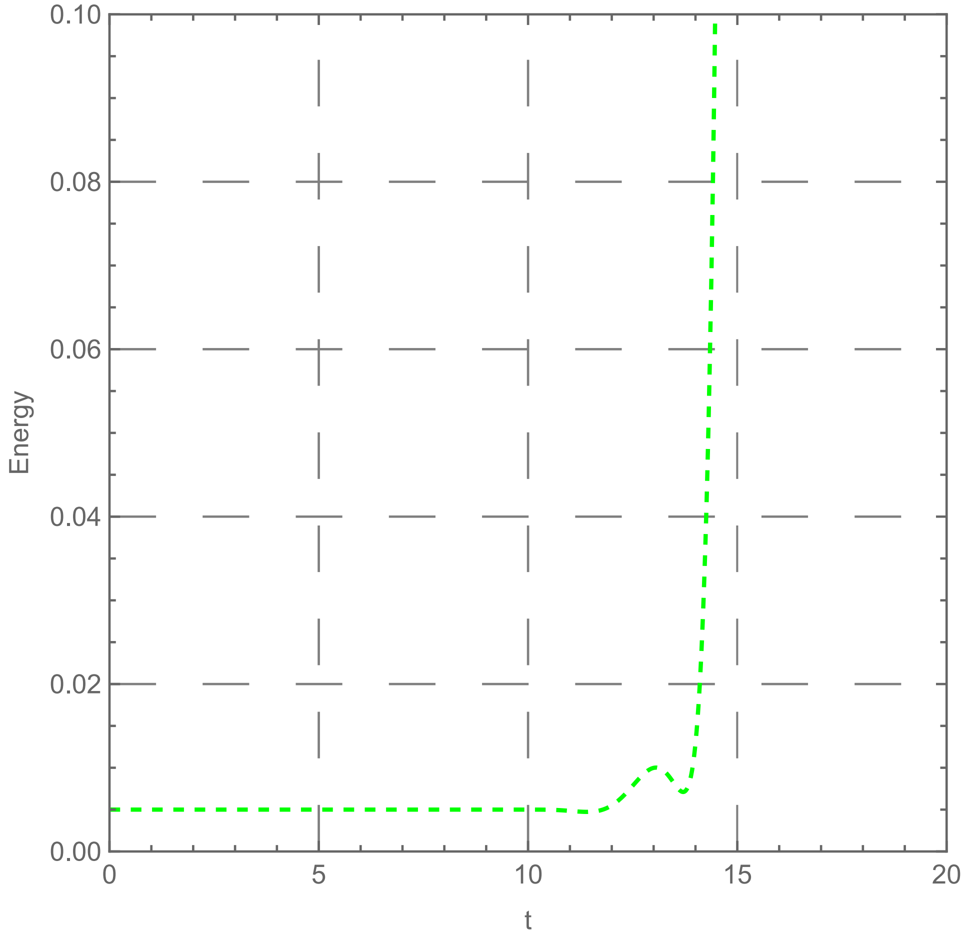




What could possibly go wrong?

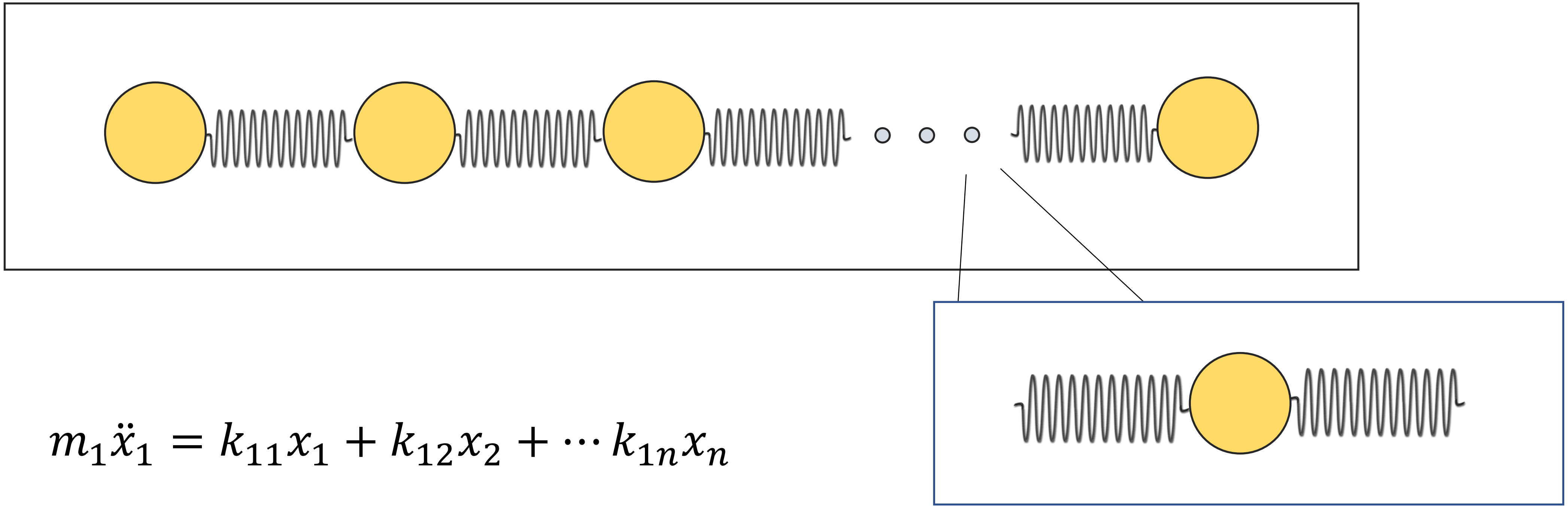

0.00500000
0.00500000
0.00499999
0.00499999
0.00499998
0.00499998
0
2
4
0.010
0.008
0.006
0.004
0.002
0.000
9.0
9.5
10.0
10.5
11.0
11.5
12.0
t
Energy
0.008
0.006
0.004
0.002
0.000
0
2
4
6
8
10
t
Energy





Truncation error always will be exposed after sufficient iterations

Moving onto the spring chain

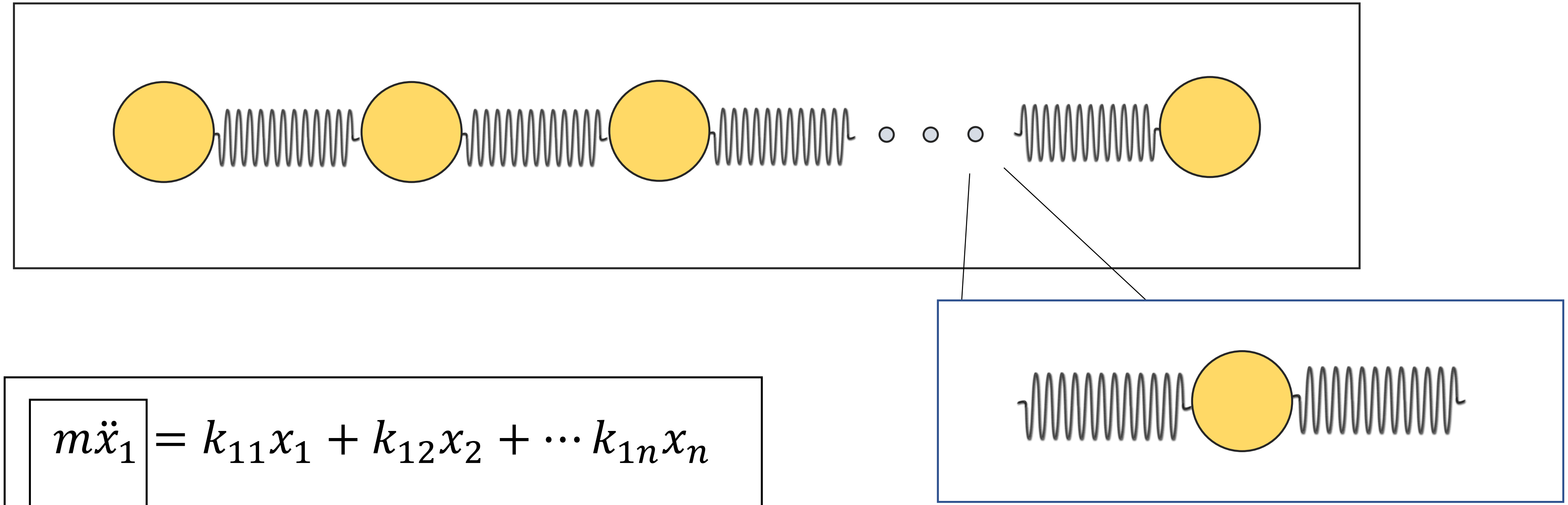


$$m_1\ddot{x}_1 = k_{11}x_1 + k_{12}x_2 + \cdots k_{1n}x_n$$

$$m_2\ddot{x}_2 = k_{21}x_1 + k_{22}x_2 + \cdots k_{2n}x_n$$

∘ ∘ ∘

$$m_n\ddot{x}_n = k_{n1}x_1 + k_{n2}x_2 + \cdots k_{nn}x_n$$



$$m\ddot{x}_1 = k_{11}x_1 + k_{12}x_2 + \cdots k_{1n}x_n$$

$$m\ddot{x}_2 = k_{21}x_1 + k_{22}x_2 + \cdots k_{2n}x_n$$

$$\circ \quad \circ \quad \circ$$

$$m\ddot{x}_n = k_{n1}x_1 + k_{n2}x_2 + \cdots k_{nn}x_n$$

Matrix form

$$m\begin{pmatrix} \ddot{x}_1 \\ \ddot{x}_2 \\ \ddot{x}_3 \\ \cdots \\ \ddot{x}_n \end{pmatrix} = \begin{pmatrix} k_{11} & \cdots & k_{n1} \\ \vdots & \ddots & \vdots \\ k_{1n} & \cdots & k_{nn} \end{pmatrix}\begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ \cdots \\ x_n \end{pmatrix}$$

Small aid from Linear Algebra:

Matrix form

$$
m\begin{pmatrix} \ddot{x}_1 \\ \ddot{x}_2 \\ \ddot{x}_3 \\ \cdots \\ \ddot{x}_n \end{pmatrix} = \begin{pmatrix} k_{11} & \cdots & k_{n1} \\ \vdots & \ddots & \vdots \\ k_{1n} & \cdots & k_{nn} \end{pmatrix}\begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ \cdots \\ x_n \end{pmatrix}
$$

$$
m\,\ddot{X} = K\,X
$$

Let's search solution in form of:

$$
X(t) = X_0 e^{i\omega t}
$$

Characteristic equation:

$$
-\omega^2 m\, X_0 e^{i\omega t} = K X_0 e^{i\omega t}
$$

Or simply we have eigenvalue problem:

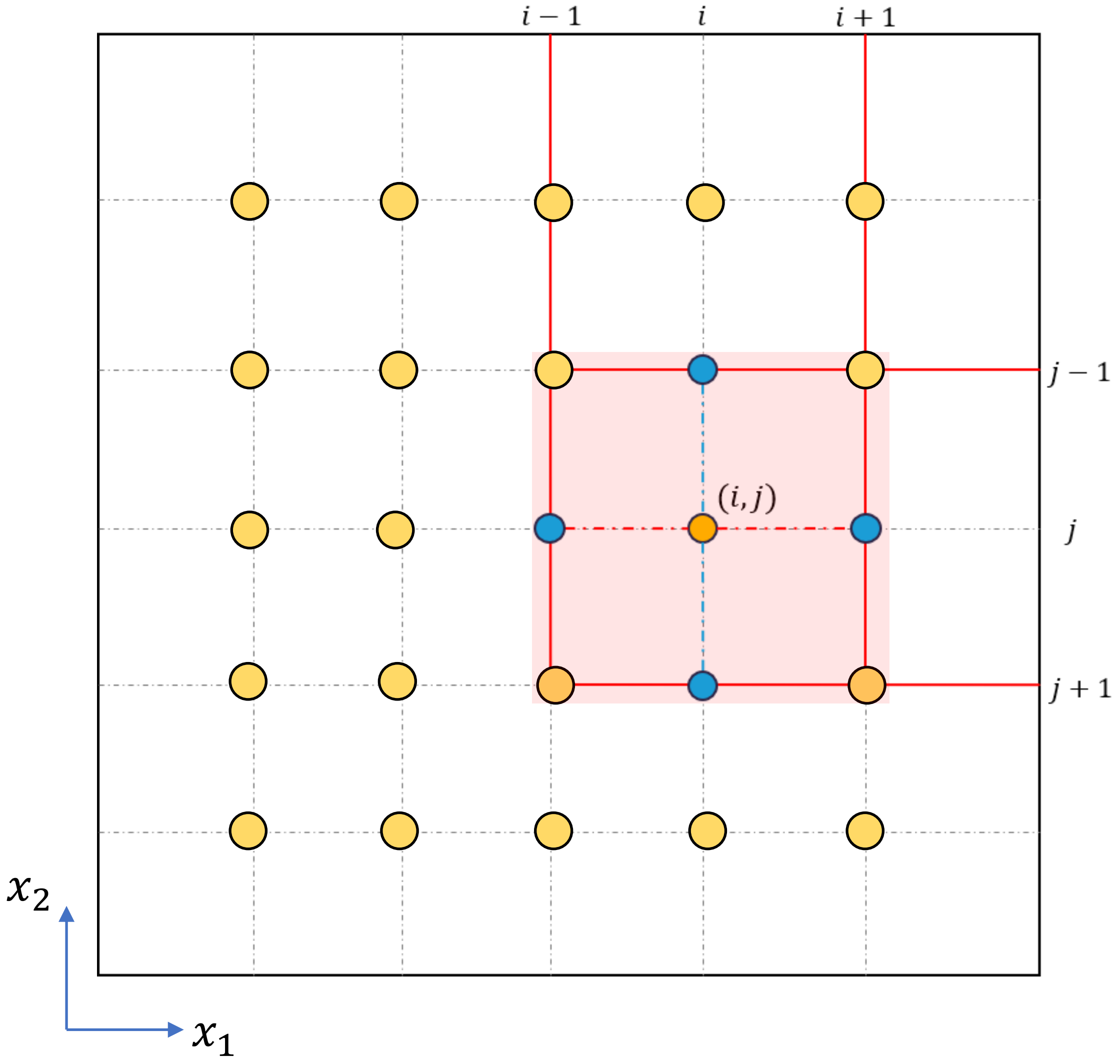
$$
(K + \omega^2 m I)X = 0
$$

$$
\boxed{\det(K + \omega^2 m I) = 0}
$$

$$
\det(K + \omega^2 m I) = 0
$$

I dare you to solve this set manually

2D Square lattice



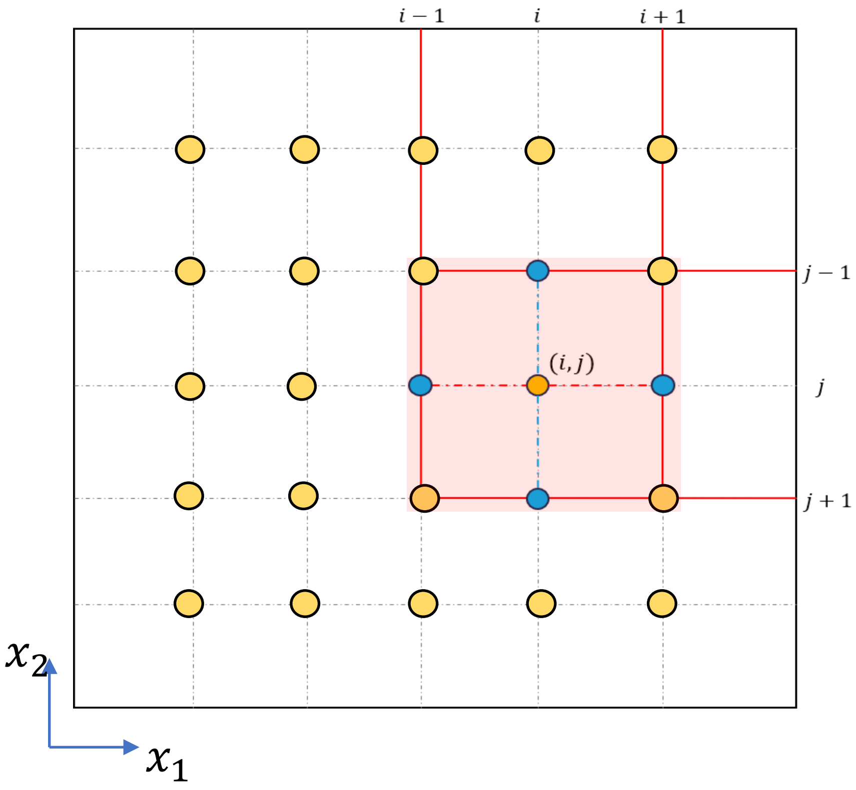
Discretization saves the day:

For (i,j) particle:

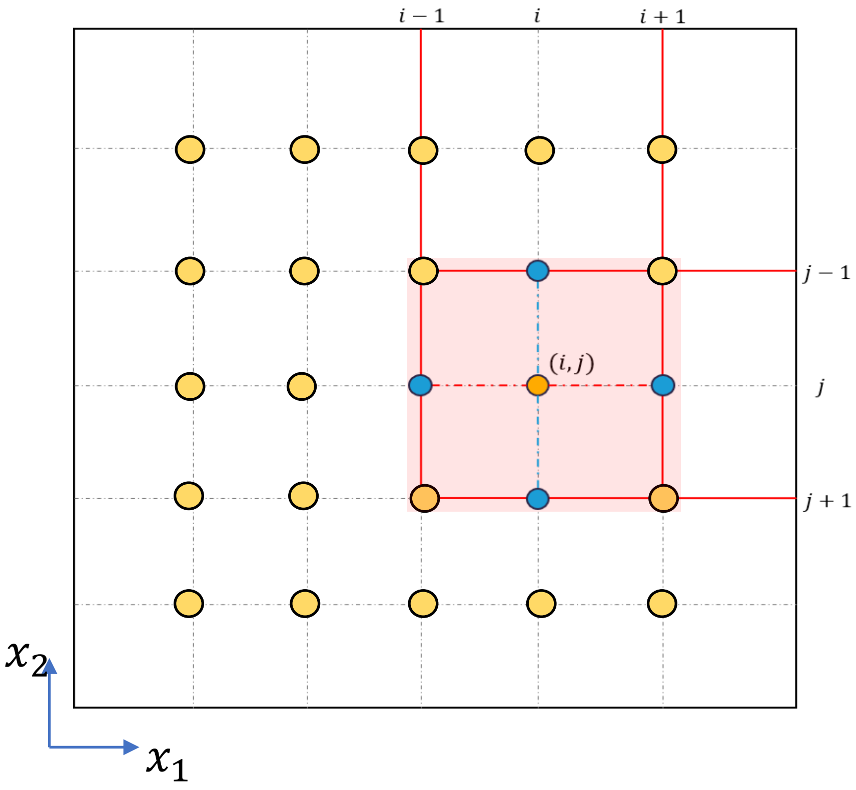
$$m\ddot{x}_{ij} = -k(x_{ij+1} - x_{ij}) + k(x_{ij} - x_{i-1j})$$
$$= k(x_{ij+1} + x_{ij-1} - 2x_{ij})$$

$$m\ddot{x}_{ij} = k(x_{ij+1} + x_{ij-1} - 2x_{ij})$$

But in our case we can always simulate piecewise way!!

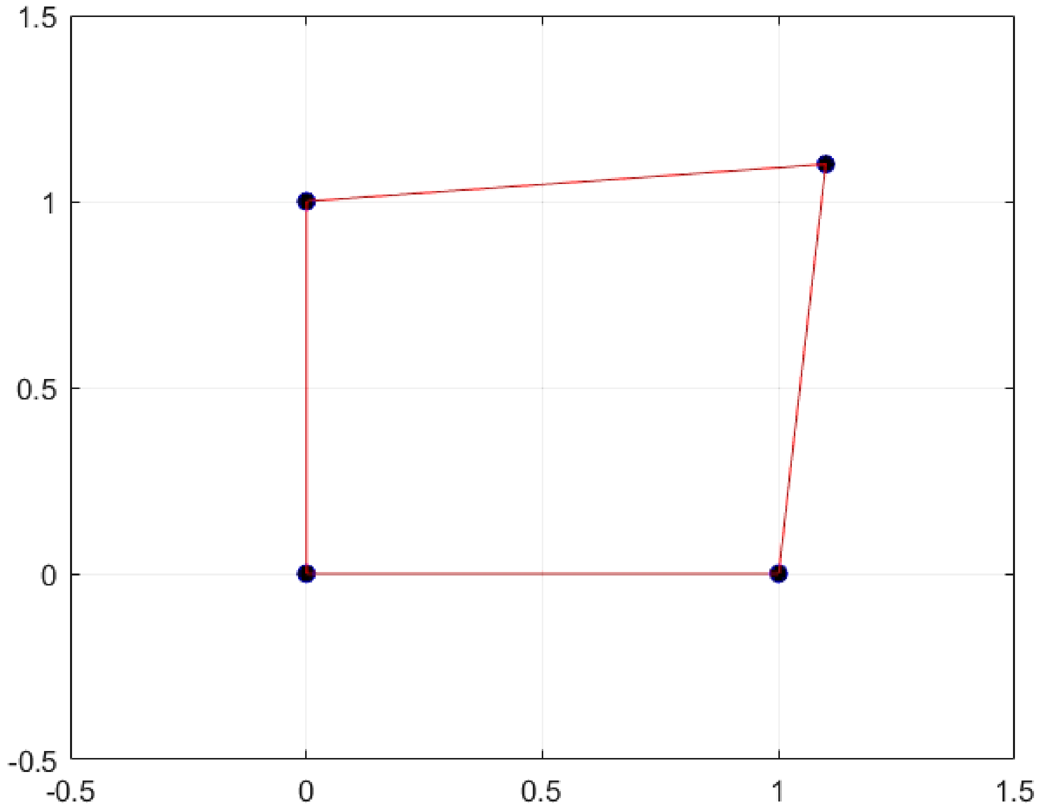


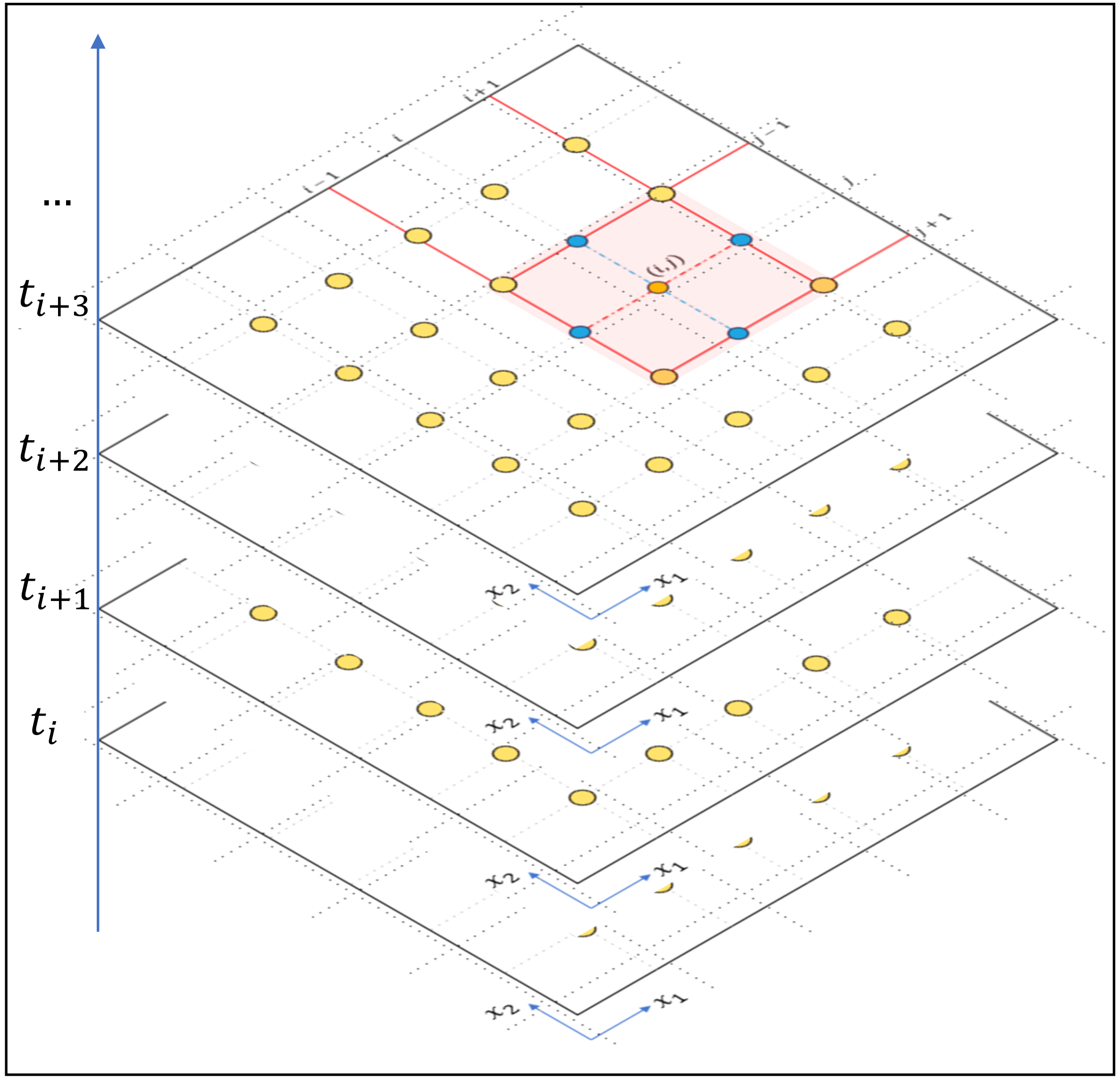

```
% lattice constants:
M=1;
k=100;
% distance and size:
lattice_width=5;
lattice_height=5;
l0=1;
xdis=1;
ydis=1;

%initial Coordinates
x_matrix=zeros(lattice_height,lattice_width);
y_matrix=zeros(lattice_height,lattice_width);

%initial velocities
vx_matrix=zeros(lattice_height,lattice_width);
vy_matrix=zeros(lattice_height,lattice_width);

%masses on the lattice
mass_matrix=zeros(lattice_height,lattice_width);
```




```matlab
% Horizontal springs (Runge-Kutta 4th order)
for a = 1:lattice_height
    for b = 1:lattice_width - 1
        % Calculate forces using RK4 for horizontal springs
        % Initial orientation and displacement vector
        orient = [x_matrix(a, b+1) - x_matrix(a, b), y_matrix(a, b+1) - y_matrix(a, b)];
        norm_orient = norm(orient);

        % Calculate intermediate forces for RK4
        ac1_k1 = (1 - l0 / norm_orient) * orient * k / mass_matrix(a, b);
        ac2_k1 = -(1 - l0 / norm_orient) * orient * k / mass_matrix(a, b+1);

        % Half step for k2
        orient_k2 = orient + 0.5 * dt * ac1_k1;
        norm_orient_k2 = norm(orient_k2);
        ac1_k2 = (1 - l0 / norm_orient_k2) * orient_k2 * k / mass_matrix(a, b);
        ac2_k2 = -(1 - l0 / norm_orient_k2) * orient_k2 * k / mass_matrix(a, b+1);

        % Half step for k3
        orient_k3 = orient + 0.5 * dt * ac1_k2;
        norm_orient_k3 = norm(orient_k3);
        ac1_k3 = (1 - l0 / norm_orient_k3) * orient_k3 * k / mass_matrix(a, b);
        ac2_k3 = -(1 - l0 / norm_orient_k3) * orient_k3 * k / mass_matrix(a, b+1);

        % Full step for k4
        orient_k4 = orient + dt * ac1_k3;
        norm_orient_k4 = norm(orient_k4);
        ac1_k4 = (1 - l0 / norm_orient_k4) * orient_k4 * k / mass_matrix(a, b);
        ac2_k4 = -(1 - l0 / norm_orient_k4) * orient_k4 * k / mass_matrix(a, b+1);

        % Update velocities using RK4 weighted sum
        vx_matrix(a, b) = vx_matrix(a, b) + dt * (ac1_k1(1) + 2*ac1_k2(1) + 2*ac1_k3(1) + ac1_k4(1)) / 6;
        vy_matrix(a, b) = vy_matrix(a, b) + dt * (ac1_k1(2) + 2*ac1_k2(2) + 2*ac1_k3(2) + ac1_k4(2)) / 6;

        vx_matrix(a, b+1) = vx_matrix(a, b+1) + dt * (ac2_k1(1) + 2*ac2_k2(1) + 2*ac2_k3(1) + ac2_k4(1)) / 6;
        vy_matrix(a, b+1) = vy_matrix(a, b+1) + dt * (ac2_k1(2) + 2*ac2_k2(2) + 2*ac2_k3(2) + ac2_k4(2)) / 6;
    end
end
```

Single cell simulation





References:

- **Matsumura, Syo. (2019). Atom locations in a Ni doped η-(Cu,Ni)6Sn5 intermetallic compound. Scripta Materialia. 158. 1-5. 10.1016/j.scriptamat.2018.08.020.**

- **Lipson, R. & Lu, C. (2009). Photonic crystals: A unique partnership between light and matter. EUROPEAN JOURNAL OF PHYSICS Eur. J. Phys. 30. 33-48. 10.1088/0143-0807/30/4/S04.**

Suggested material to check out:

https://lampz.tugraz.at/~hadley/ss1/phonons/1d/1dphonons.php

Until we meet again?

雲 です が、 なに か？

Appendix