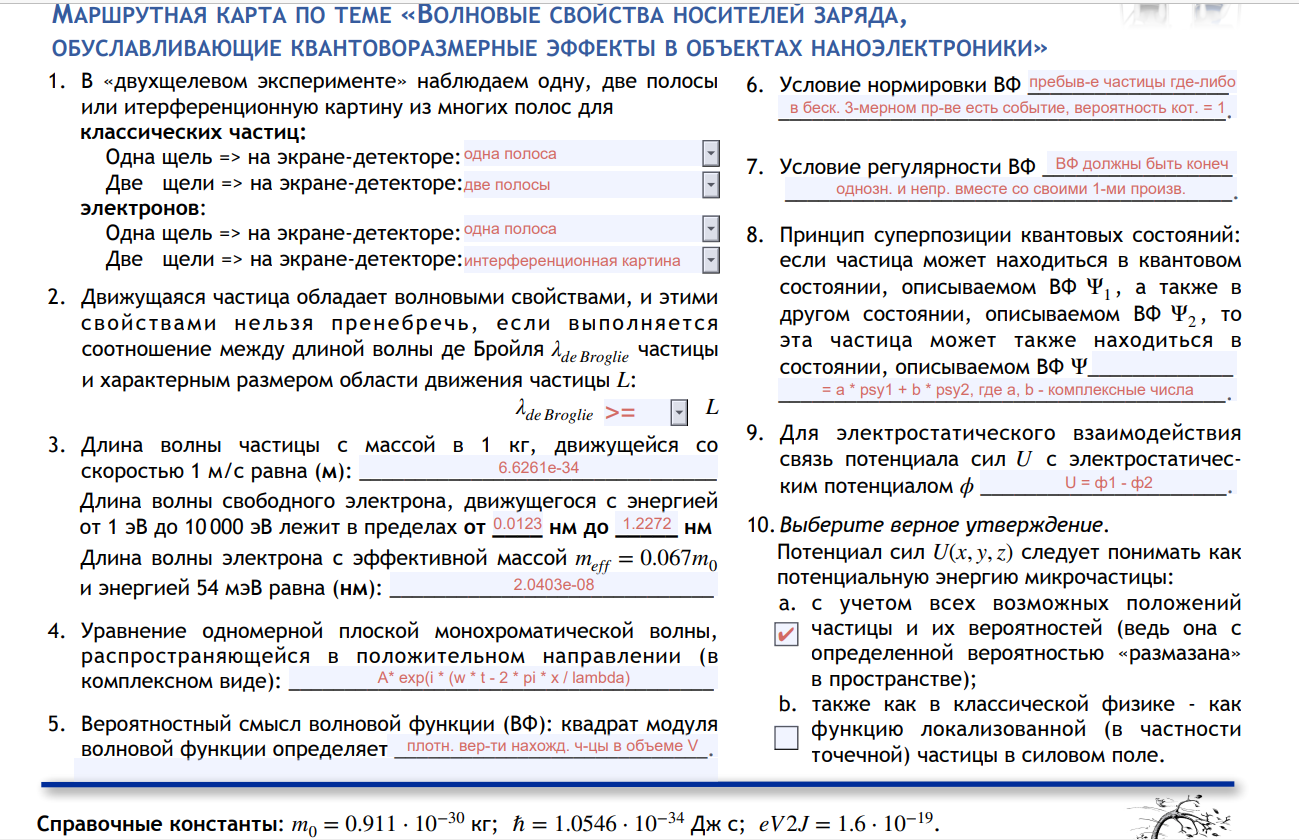
L01\_20230902

Цель: изучение основ квантовой физики

clc, clear, close; %Task1

h = 6.6261 \* 10^-34; m = 1; v = 1; p = m \* v; lambda1 = h / p %Task2

m0 = 0.911 \* 10^-30; e = [1\*1.6\*10^-19; 10000\*1.6\*10^-19]; lambda2 = [h/sqrt(2\*m0\*e(1))\*10^9; h/sqrt(2\*m0\*e(2))\*10^9] %Task3

e1 = 54 \* 1.6 \* 10^-19 \* 10^-3; meff = 0.067 \* m0; h1 = 1.0546 \* 10^-34; lambda3 = 2 \* pi \* h1 / sqrt(2 \* e1 \* meff)

datestr(clock)

*lambda1 = 6.6261e-34*

*lambda2 =*

*1.2272*

*0.0123*

*lambda3 = 2.0403e-08*

*ans = '10-Sep-2023 20:52:35'*

*Published with MATLAB® R2023a*

1

L01\_20220214\_wave

Цель: изучение работы с анимацией

clc, clear, close all

x = 0:0.01:2;

x\_dot = 0:0.2:2;

drawArrow = @(x,y,varargin) quiver( x(1),y(1),x(2)-x(1),y(2)-y(1),0, varargin{:} )

while 1

for i = 0:0.05:2

plot(x, sin(2\*pi\*x - pi\*i), x\_dot, sin(2\*pi\*x\_dot - pi\*i), '\*r')

hold on

for ar = 1:11

x1 = [x\_dot(ar) x\_dot(ar)];

A = 1 - abs(sin(2\*pi\*x\_dot(ar) - pi\*i));

y = sin(2\*pi\*x\_dot(ar) - pi\*i);

y1 = [y, y - A \* cos(2\*pi\*x\_dot(ar) - pi\*i)];

drawArrow(x1, y1, 'linewidth', 1, 'color', 'k')

end

hold off

xlim([0 2])

ylim([-2 2])

frame = getframe;

image = frame2im(frame);

[X, cmap] = rgb2ind(image, 256);

if i == 0

imwrite(X, cmap, 'myanim.gif', 'gif', 'LoopCount', Inf, ...

'DelayTime', 1/24);

else

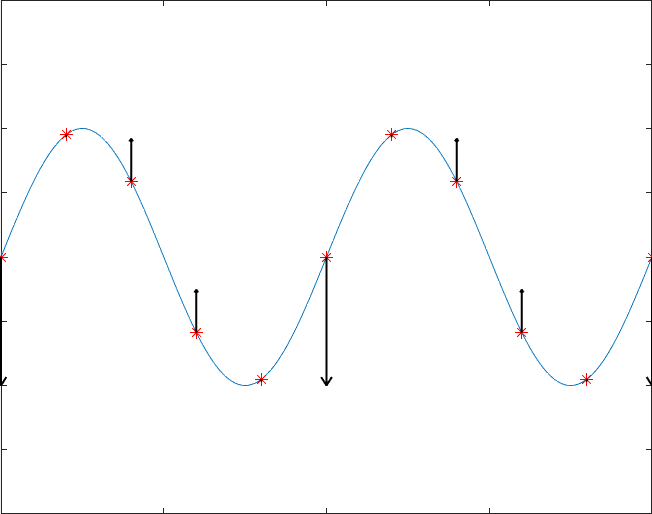
imwrite(X, cmap, 'myanim.gif', 'gif', 'WriteMode', 'append', ...

'DelayTime', 1/24);

end

end

end



L01\_20230916\_psi2d

Цель: изучение волновых процессов

clc, clear, close;

syms A x a b;

% source function f = A\*exp(-x^2/a^2 + 1i\*b\*x); % my real part f1 = A\*exp(-x^2/a^2)\*cos(b\*x); % gauss = (A\*exp(-x^2/a^2)\*cos(b\*x))^2 + (A\*exp(-x^2/a^2)\*sin(b\*x))^2; gauss\_simple = (A\*exp(-x^2/a^2))^2;

A = 5; b = 2 \* pi \* 4 / 5; a = sqrt(2 \* (7 / 6)^2);

x = -5:0.084:5;

% plot from my real part y1 = subs(f1);

% plot using real() y = subs(f); y2 = real(y);

yg = subs(gauss\_simple);

figure1 = figure; plot(x, y1, 'b', x, y2, 'r.'); title('$\Re \left[ \Psi(x,0) = A \exp \left( -\frac{x^2}{a^2} + \imath bx \right) \right]$','Interpreter','latex') xlabel('x'); ylabel('Analytical $\Re(\Psi)$ \& real($\Psi$)','Interpreter','latex'); legend('$\Re(\Psi)$','real($\Psi$)','Interpreter','latex');

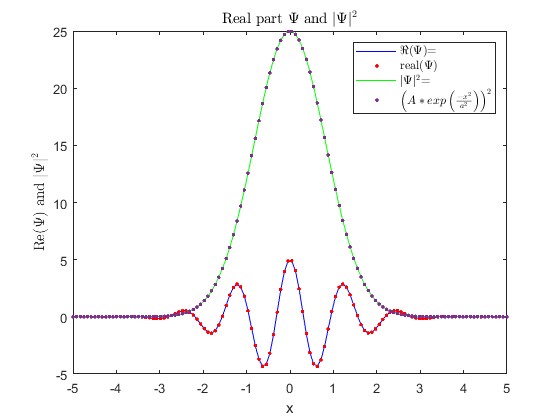
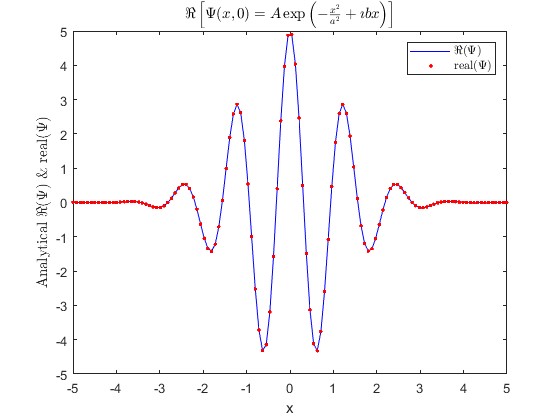
figure2 = figure; plot(x, y1, 'b', x, y2, 'r.', x, yg, 'g', x, yg, '.'); title('Real part $\Psi$ and $|\Psi|^2$','Interpreter','latex') xlabel('x'); ylabel('Re($\Psi$) and $|\Psi|^2$','Interpreter','latex'); legend('$\Re(\Psi)$=','real($\Psi$)','$|\Psi|^2$=','$\left(A\*exp\left(\frac{x^2}{a^2}\right)\right)^2$','Interpreter','latex');

datetime(clock)

*ans =*

*datetime*

*07-Oct-2023 12:11:20*



*Published with MATLAB® R2023a*

Отчёт:

Аналитическое получение действительной части:

Psi = A\*exp(-x^2/a^2 + i\*b\*x) = A\*exp(-x^2/a^2)\*exp(i\*b\*x)

exp(i\*b\*x) = cos(b\*x) + i\*sin(b\*x)

Psi = A\*exp(-x^2/a^2)\*cos(b\*x) + i\*A\*exp(-x^2/a^2)\*sin(b\*x), где первое слагаемое содержит действительную часть, второе – мнимую.

Таким образом,

A\*exp(-x^2/a^2)\*cos(b\*x) – действительная часть данной пси-функции

Получение |Psi|^2:

(A\*exp(-x^2/a^2)\*cos(b\*x))^2 + (A\*exp(-x^2/a^2)\*sin(b\*x))^2

Это выражение необходимо сократить. Вынесем A\*exp(-x^2/a^2)^2:

A\*exp(-x^2/a^2)^2 \* (cos(b\*x)^2 + sin(b\*x)^2) = A\*exp(-x^2/a^2)^2 \* 1

L02\_20230916

Цель: освоение методов построения графиков в среде MATLAB, построение графиков сложных функций

clear, clc, close; syms x y; s = solve(x^2+(y\*5/4-sqrt(abs(x)))^2 == 1, y); x = -1:0.01:1; y = subs(s);

% Create figure figure1 = figure;

% Create axes axes1 = axes('Parent',figure1); hold(axes1,'on');

% Create multiple line objects using matrix input to plot plot(x, y,'LineWidth',3,'Color',[1 0 0]); fill(x, y, 'b')

% Create ylabel ylabel('Y');

% Create xlabel xlabel('X');

% Create title txt = '$x^2+\left(\frac{5y}{4}-\sqrt{|x|}\right)^2 = 1$'; title(txt,'Interpreter','latex');

box(axes1,'on'); hold(axes1,'off'); % Set the remaining axes properties set(axes1,'Color',[0.650980392156863 0.650980392156863 0.650980392156863],... 'FontAngle','italic','XColor',[0 0 0],'XGrid','on','YColor',[0 0 0],'YGrid',... 'on','ZColor',[0 0 0], Layer = 'top');

figure2 = figure;

% L subplot(2,2,1) x = 0:0.1:3; y = 1./x; plot(x, y); txt = '$y = \frac{1}{x}$'; title(txt,'Interpreter','latex');

% O subplot(2,2,2) syms x y; s2 = solve(x^2 + y^2 == 9, y); x = -3:0.1:3; y = subs(s2);

plot(x, y); txt = '$x^2 + y^2 = 9$'; title(txt,'Interpreter','latex');

% V subplot(2,2,3) x = -5:0.1:5; y = abs(-2\*x); plot(x, y); txt = '$y = |-2x|$'; title(txt,'Interpreter','latex');

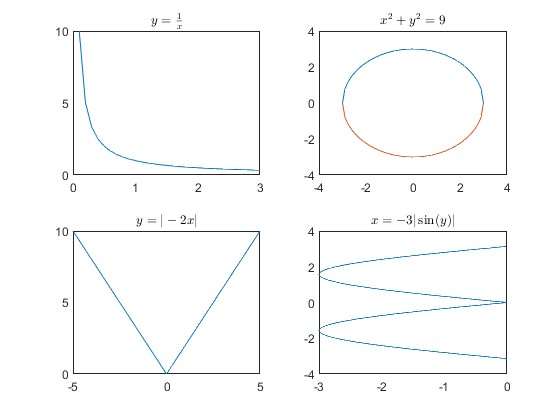
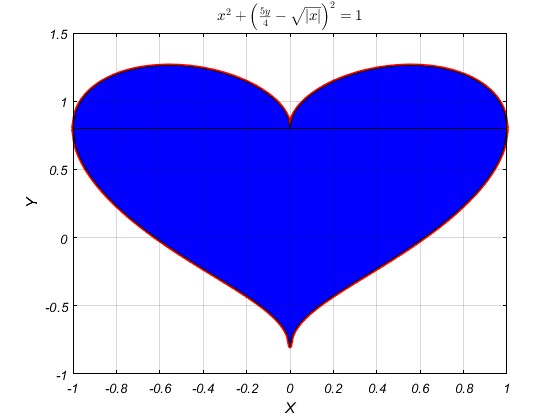
% E subplot(2,2,4) y = -pi:0.01:pi; x = -3 \* abs(sin(y)); plot(x, y); txt = '$x = -3|\sin(y)|$'; title(txt,'Interpreter','latex');

datetime(clock)

*ans =*

*datetime*

*23-Sep-2023 14:48:24*



*Published with MATLAB® R2023a*

L03\_20230930

Цель: освоение методов работы с функциями в среде MATLAB, изучение потенциальной ямы, решение уравнения Шредингера

function [En\_meV, wn] = EnergyFrequency(meff, L, n) load('constants.mat', 'hbar', 'm0', 'J2eV')

En\_J = (hbar \* pi \* n / (L \* 1e-9)).^2 ./ (2 \* meff \* m0); En\_meV = En\_J \* J2eV \* 1e3;

wn = En\_J / hbar;

disp(['For an electron meff = ', num2str(meff), ', in L = ', num2str(L), ' nm: '])

fprintf('E%li = %3i meV; w%li = %1.0e rad/s\n', [n; round(En\_meV); n; wn]) end

Not enough input arguments.

Error in EnergyFrequency (line 3)

En\_J = (hbar \* pi \* n / (L \* 1e-9)).^2 ./ (2 \* meff \* m0);

*Published with MATLAB® R2023a*

clc, clear, close all load('constants.mat')

deltaEnergy = @ (meff, L, n) round((hbar \* pi / (L \* 1e-9))^2 / (2 \* meff \* m0) \* J2eV \* 1e3 \* (2 \* n + 1));

fprintf('deltaE%i = %i meV\n', [12; deltaEnergy(0.07, 10, 1)]);

fprintf('deltaE%i = %i meV\n', [23; deltaEnergy(0.07, 10, 2)]);

% n = 1

Lmax = @ (meff, T) round(hbar \* pi \* sqrt((2 \* 1 + 1) / (3 \* meff \* 1.38e-23 \* m0 \* T)) \* 10^9);

fprintf('For practical systems\n(meff = %.2f, T = %d K):\nLmax = %d nm\n', [0.07; 300; Lmax(0.07, 300)])

datetime(clock)

*deltaE12 = 161 meV deltaE23 = 269 meV For practical systems*

*(meff = 0.07, T = 300 K):*

*Lmax = 20 nm ans =*

*datetime*

*11-Oct-2023 21:02:43*

*Published with MATLAB® R2023a*

clc, clear, close all load constants.mat

% Handle-functions

Energy = @ (n, L, meff) (hbar \* pi \* n / (L \* 1e-9))^2 / (2 \* meff \* m0) \* J2eV \* 1e3;

% Energy levels figure

hold on

meff = 0.07;

L = 10;

E1 = Energy(1, L, meff); for n = 1:3

E = Energy(n, L, meff); x = [-L / 2; L / 2];

y = [E / E1; E / E1];

plot(x, y) end

Plot(gca, L, 1); hold off

% Solve

figure hold on L = 10;

x = -L/2:0.1:L/2;

for n = 1:3

Psin = sqrt(2 / L) \* (sin(pi \* n \* x / L + pi \* n / 2)); plot(x, Psin)

end

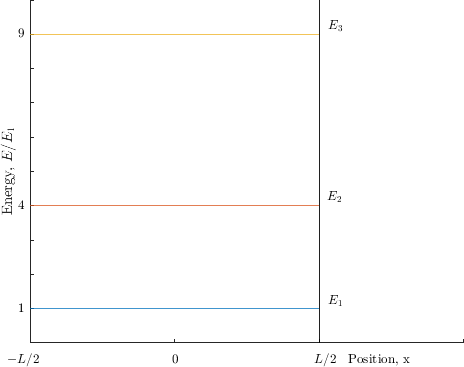
Plot(gca, L, 0); hold off

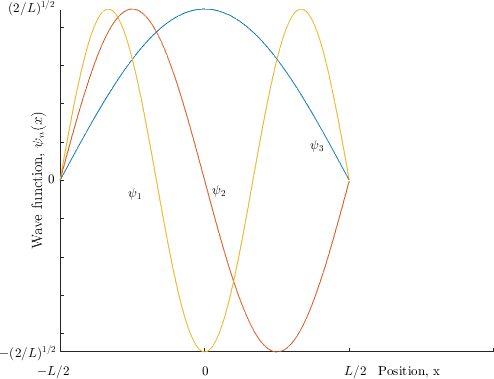
datetime(clock)

*ans =*

*datetime*

*11-Oct-2023 19:54:30*





*Published with MATLAB® R2023a*

function [gca] = Plot(gca, L, key) if key == 0

yt = get(gca,'YTick'); set(gca,'YTickLabel', sprintf(' ',yt)) xt = get(gca,'XTick'); set(gca,'XTickLabel', sprintf(' ',xt)) xlim([-5, 10])

ylim([-(2/L)^(1/2), (2/L)^(1/2)])

ylabel('Wave function, $\psi\_{n}(x)$', 'Interpreter', 'latex')

text(-2.6613, -0.0357, '$\psi\_{1}$', 'Interpreter', 'latex')

text(0.265, -0.0270, '$\psi\_{2}$', 'Interpreter', 'latex') text(3.6521, 0.0896, '$\psi\_{3}$', 'Interpreter', 'latex')

% x

text(-(L / 2) - 0.8, -0.5, '$-L/2$', 'Interpreter', 'latex')

text((L / 2) - 0.2, -0.5, '$L/2$', 'Interpreter', 'latex')

text(-0.1, -0.5, '$0$', 'Interpreter', 'latex')

text(L/2 + 1, -0.5, 'Position, x', 'Interpreter', 'latex')

% y

text(-(L / 2) - 2.1, -0.45, '$-(2/L)^{1/2}$', 'Interpreter', 'latex')

text(-(L / 2) - 0.4, 0, '$0$', 'Interpreter', 'latex')

text(-(L / 2) - 1.8, 0.45, '$(2/L)^{1/2}$', 'Interpreter', 'latex')

else

ylim([0; 10])

xlim([-5, 10])

ylabel('Energy, $E/E\_{1}$', 'Interpreter', 'latex') yt = get(gca,'YTick');

set(gca,'YTickLabel', sprintf(' ',yt)) xt = get(gca,'XTick'); set(gca,'XTickLabel', sprintf(' ',xt))

plot([L/2 L/2], get(gca, 'Ylim'), 'k');

|  |  |  |
| --- | --- | --- |
| text(5.3111, | 9.2315, | '$E\_3$', 'Interpreter', 'latex') |
| text(5.2650, | 4.2510, | '$E\_2$', 'Interpreter', 'latex') |
| text(5.2880,  % x | 1.2160, | 9, '$E\_1$', 'Interpreter', 'latex') |

text(-(L / 2) - 0.8, -0.5, '$-L/2$', 'Interpreter', 'latex')

text((L / 2) - 0.2, -0.5, '$L/2$', 'Interpreter', 'latex')

text(-0.1, -0.5, '$0$', 'Interpreter', 'latex')

text(L/2 + 1, -0.5, 'Position, x', 'Interpreter', 'latex')

% y

text(-(L / 2) - 0.4, 1, '1', 'Interpreter', 'latex')

text(-(L / 2) - 0.4, 4, '4', 'Interpreter', 'latex')

text(-(L / 2) - 0.4, 9, '9', 'Interpreter', 'latex')

end end

Not enough input arguments.

Error in Plot (line 2) if key == 0

*Published with MATLAB® R2023a*

clc, clear, close all load constants.mat

EnergyFrequency(0.07, 20, 2); datetime(clock)

*For an electron meff = 0.07, in L = 20 nm:*

*E2 = 54 meV; w2 = 8e+13 rad/s ans =*

*datetime*

*11-Oct-2023 21:27:07*

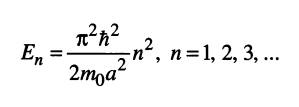
*Published with MATLAB® R2023a*

Отчёт

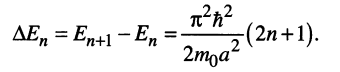
Часть 1

Расчёт Lmax производился из следующих соображений:

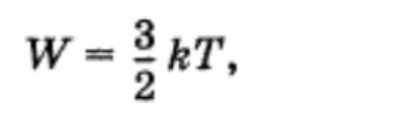
Полная энергия частицы, движущейся в потенциальной яме с бесконечно высокими стенками:



Разность значений энергий n-го и (n+1) энергетических уровней:



Энергия теплового движения:

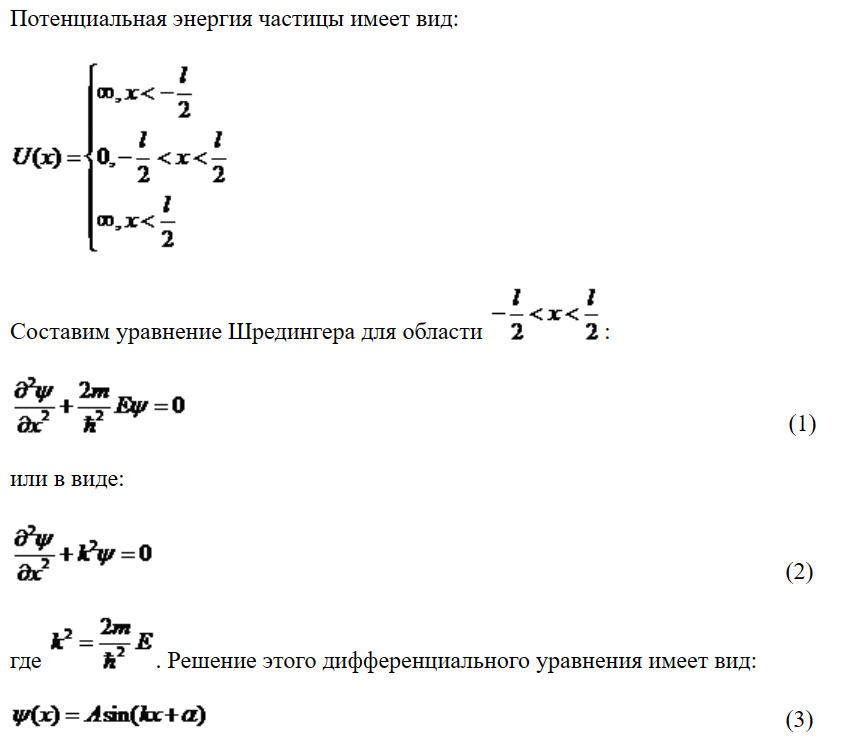


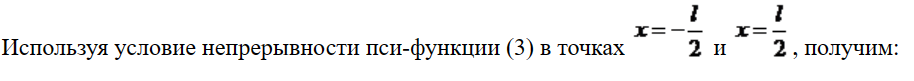


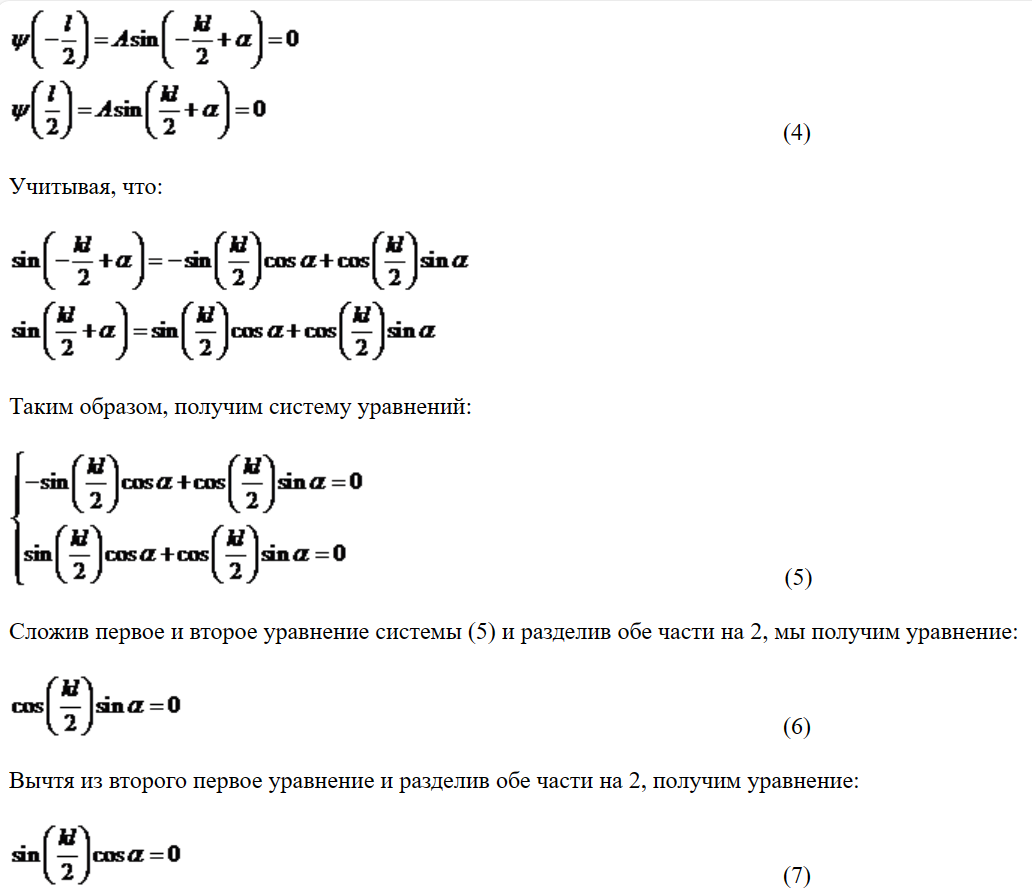
И отсюда находим а, который и является Lmax.

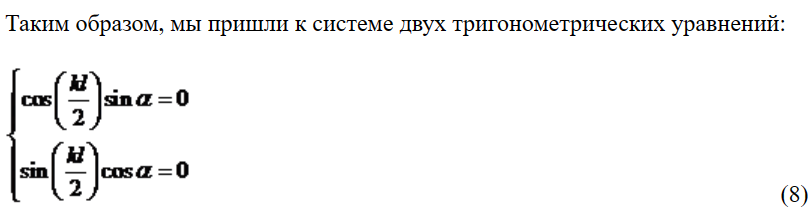
Часть 2

Вывод решения уравнения Шредингера для потенциальной ямы с бесконечно высокими стенками с началом отсчёта в центре ямы:

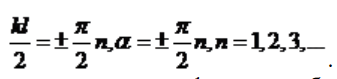




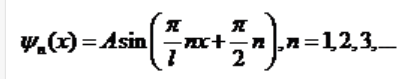




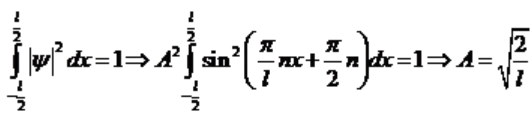
Из этой системы:



Тогда пси-функции собственных состояний имеют вид:

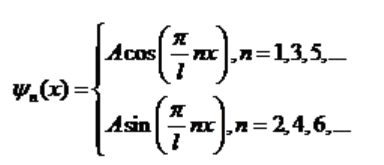


Здесь можно получить A:



Графики данной функции для n = 1, 2, 3 были построены в MainFunction.

Эту функцию также можно переписать в виде:



Но тогда могут возникнуть проблемы при построении:

-------------------------------------------------------------------------------------

figure

hold on

L = 10;

x = -L/2:0.1:L/2;

for n = 1:3

if mod(n, 2) == 0

Psin = sqrt(2 / L) \* (sin(pi \* n \* x / L));

plot(x, Psin)

else

Psin = sqrt(2 / L) \* (cos(pi \* n \* x / L));

plot(x, Psin)

end

end

Plot(gca, L);

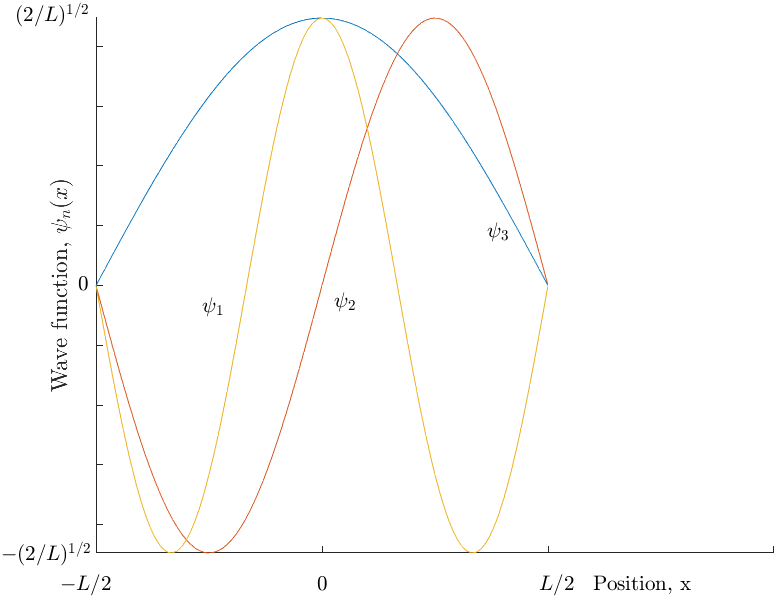
text(-2.6613, -0.0357, '$\psi\_{1}$', 'Interpreter', 'latex')

text(0.265, -0.0270, '$\psi\_{2}$', 'Interpreter', 'latex')

text(3.6521, 0.0896, '$\psi\_{3}$', 'Interpreter', 'latex')

hold off

-------------------------------------------------------------------------------------



Как видим, значения функций psi1 и psi2 отличаются по знаку от оригинала. Это происходит из-за того, что система  не учитывает чередование знака у синуса и косинуса, которое вызвано зависимостью фазы от n. Проследить это изменение можно руководствуясь формулами приведения. Чередование знака алгоритмически описывается следующим образом:

-------------------------------------------------------------------------------------

figure

hold on

L = 10;

x = -L/2:0.1:L/2;

change\_sin = true;

change\_cos = false;

for n = 1:5

i = 0;

if mod(n, 2) == 0

if change\_sin

Psin = sqrt(2 / L) \* (-sin(pi \* n \* x / L));

change\_sin = false;

else

Psin = sqrt(2 / L) \* (sin(pi \* n \* x / L));

change\_sin = true;

end

plot(x, Psin)

else

if change\_cos

Psin = sqrt(2 / L)\*(-cos(pi \* n \* x / L));

change\_cos = false;

else

Psin = sqrt(2 / L)\*(cos(pi \* n \* x / L));

change\_cos = true;

end

plot(x, Psin)

end

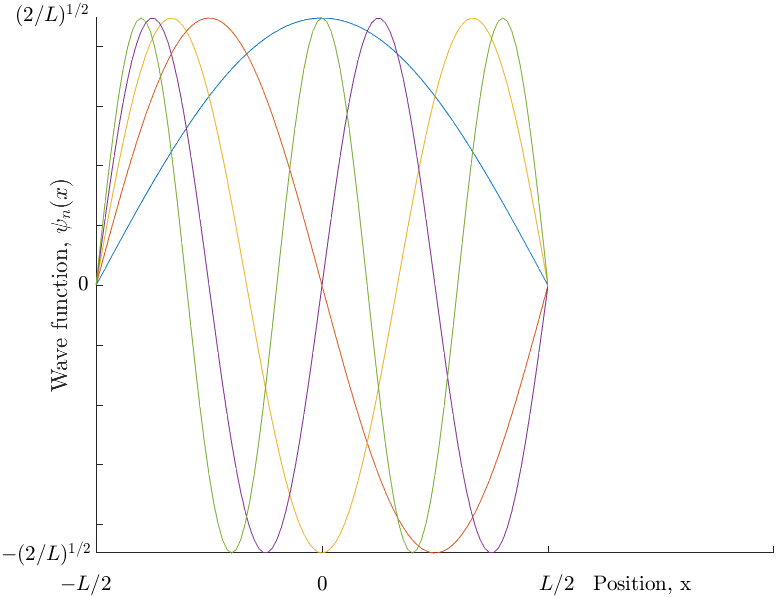
end

Plot(gca, L);

hold off

-------------------------------------------------------------------------------------

Для того, чтобы убедиться в правильности написания алгоритма, были описаны 5 уровней:



L04\_20231014

Цель: изучение численных методов на примере решения уравнения Шредингера для 1D потенциальной ямы

**Table of Contents**

........................................................................................................................................................ 1

Handle-functions ................................................................................................................................. 1 Ex.1 Analythical solve ......................................................................................................................... 1 Ex.1 Numeric solve ............................................................................................................................. 2

Ex.2 abs(psi)^2 (Analythical solve) ........................................................................................................ 3 Ex.2 abs(psi)^2 (Numeric solve) ............................................................................................................ 4 Ex.3 E~n An. and Num. solves ............................................................................................................. 5 Ex. 4 An. and Num. solves for n >> 10 .................................................................................................. 9 Ex. 5 (Advanced) En ~ n by numeric solve ........................................................................................... 11

clc, clear, close all load 'constants.mat'

# Handle-functions

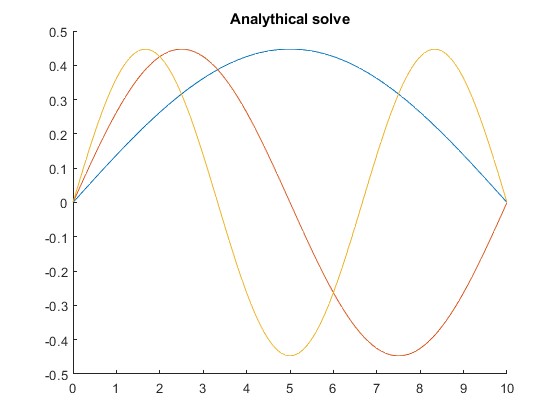
Energy = @ (n, L, meff) (hbar \* pi \* n / (L \* 1e-9))^2 / (2 \* meff \* m0) \*

J2eV \* 1e3;

# Ex.1 Analythical solve

In ex.1 we need to compare analythical and numeric solves for n = 1, 2, 3 by plotting.

figure('Name', 'Analythical solve') hold on L = 10; x = 0:0.1:L; h = 1/10; for n = 1:3 Psin = sqrt(2 / L) \* (sin(pi \* n \* x / L)); plot(x, Psin) end title('Analythical solve') hold off



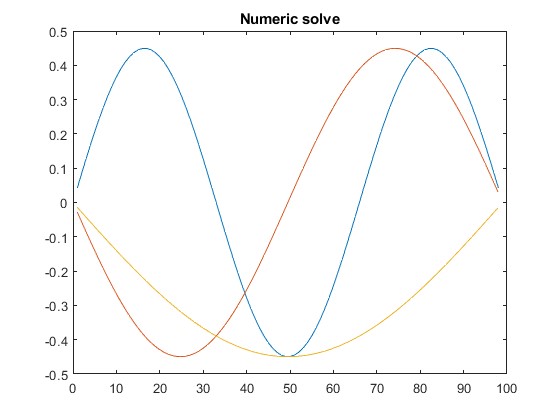
# Ex.1 Numeric solve

figure('Name', 'Numeric solve') N = 98;

I = eye(N, N); T = I; for i = 1:N for j = 1:N if i == j T(i, j) = -2; continue end

if i == j - 1 || i == j + 1 T(i, j) = 1; end end end [psi, D] = eig(T); psi = psi ./ sqrt(h); disp(sum(psi(:,N-2).^2)\*h) % Make sure numbers are correct (need to = 1) plot(1:N, psi(:,N-2), 1:N, psi(:,N-1), 1:N, psi(:,N)) title('Numeric solve')

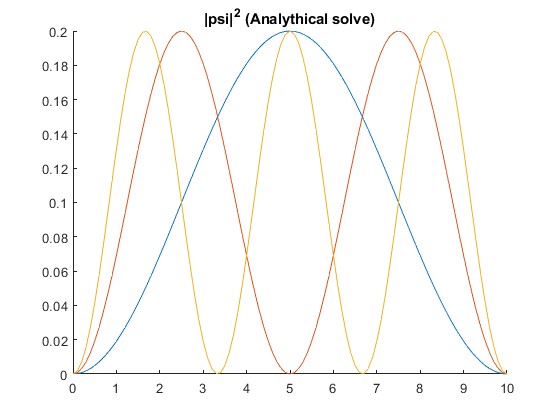
*1.0000*



# Ex.2 abs(psi)^2 (Analythical solve)

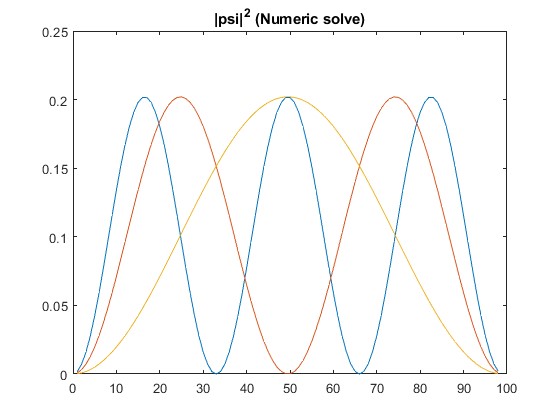
In ex.2 we need the same thing we had in ex.1 but for abs(psi)^2.

figure('Name', 'abs(psi)^2 (Analythical solve)') hold on for n = 1:3 Psin = sqrt(2 / L) \* (sin(pi \* n \* x / L)); plot(x, abs(Psin).^2) end title('|psi|^2 (Analythical solve)') hold off



# Ex.2 abs(psi)^2 (Numeric solve)

figure('Name', 'abs(psi)^2 (Numeric solve)') abPsi = abs(psi).^2; plot(1:N, abPsi(:,N-2), 1:N, abPsi(:,N-1), 1:N, abPsi(:,N)) title('|psi|^2 (Numeric solve)')



# Ex.3 E~n An. and Num. solves

In ex.3 we need to compare analythical and numeric connection between E and n by plotting. Also we need to try high n values (n = 10, 50, 90).

figure('Name', 'E~n Num. solves (n = 1...5)') E = hbar^2 \* diag(D) / (h^2 \* 2 \* m0); E = wrev(E); hold on for n = 1:5 x = [0 10]; y = [E(n)/E(1); E(n)/E(1)]; text(11, E(n)/E(1), strcat('n=', num2str(n))) plot(x, y) xlim([0 12]); end plot([10 10], get(gca, 'Ylim'), 'k'); title('E~n Num. solves (n = 1...5)') hold off

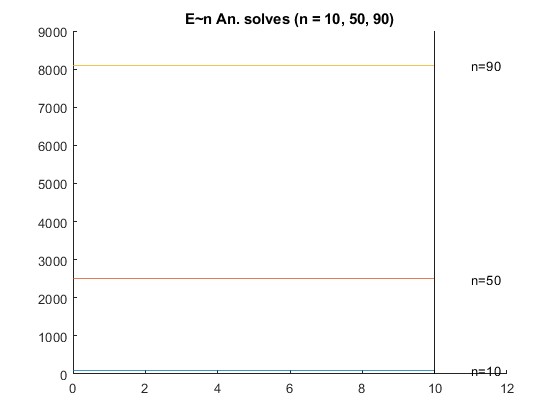
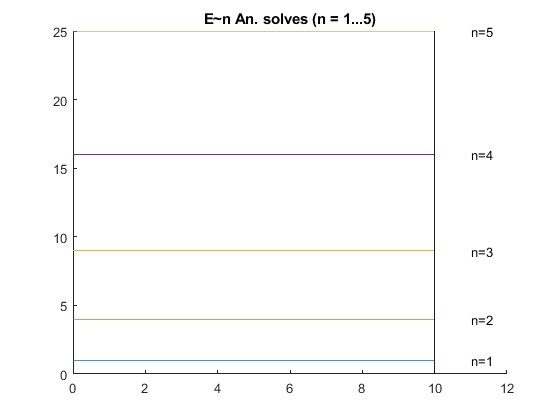
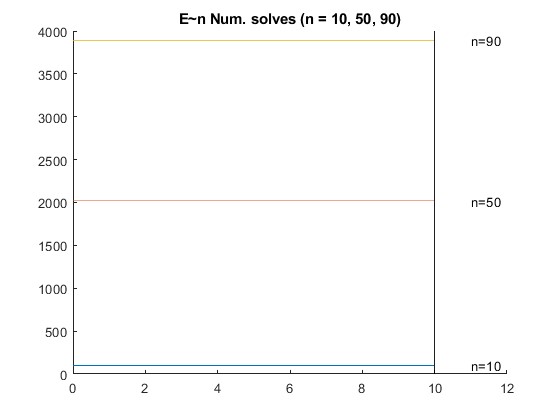
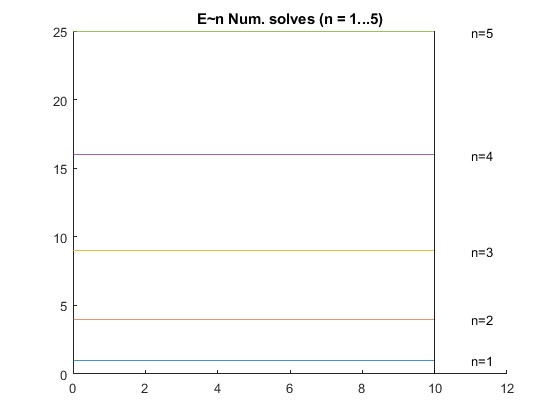
figure('Name', 'E~n Num. solves (n = 10, 50, 90)') hold on for n = [10 50 90] x = [0 10]; y = [E(n)/E(1); E(n)/E(1)]; text(11, E(n)/E(1), strcat('n=', num2str(n))) plot(x, y) xlim([0 12]); end plot([10 10], get(gca, 'Ylim'), 'k'); title('E~n Num. solves (n = 10, 50, 90)') hold off

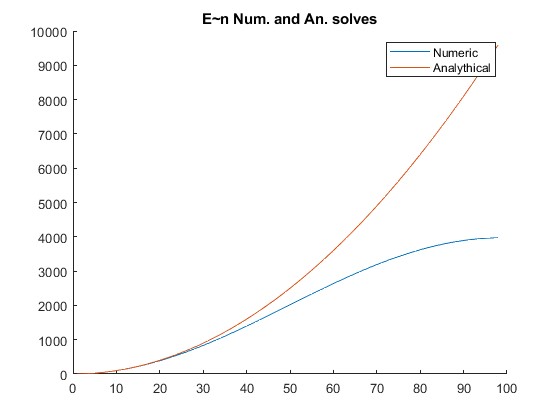
figure('Name', 'E~n An. solves (n = 1...5)') hold on for n = 1:5 E = Energy(n, 10, 0.07); E = E/Energy(1, 10, 0.07); x = [0 10]; y = [E E]; text(11, E, strcat('n=', num2str(n))) plot(x, y) xlim([0 12]); end plot([10 10], get(gca, 'Ylim'), 'k'); title('E~n An. solves (n = 1...5)') hold off

figure('Name', 'E~n An. solves (n = 10, 50, 90)') hold on for n = [10 50 90] E = Energy(n, 10, 0.07); E = E/Energy(1, 10, 0.07); x = [0 10]; y = [E E]; text(11, E, strcat('n=', num2str(n))) plot(x, y) xlim([0 12]); end plot([10 10], get(gca, 'Ylim'), 'k'); title('E~n An. solves (n = 10, 50, 90)') hold off % Another variant. All numeric and analythical energy levels.

figure('Name', 'E~n Num. and An. solves') En = hbar^2 \* diag(D) / (h^2 \* 2 \* m0); for n = 1:98 Ea(n) = Energy(n, 98, 0.07); end En = En./En(98);

En = wrev(En); Ea = Ea./Ea(1); hold on plot(1:98, En, 1:98, Ea) legend('Numeric', 'Analythical') title('E~n Num. and An. solves') hold off



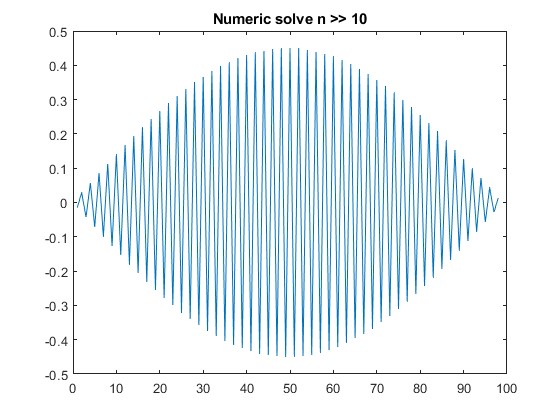
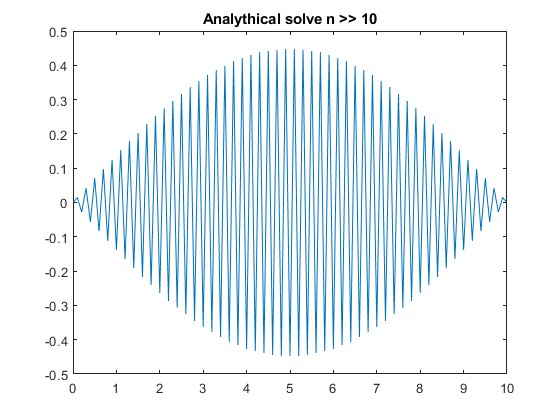


# Ex. 4 An. and Num. solves for n >> 10

In ex.4 we need the same thing we had in ex.1 but for n >> 10 (for example, the highest n we had in task).

figure('Name', 'Analythical solve n >> 10') L = 10; x = 0:0.1:L; h = 1/10; n = 99; Psin = sqrt(2 / L) \* (sin(pi \* n \* x / L)); plot(x, Psin) title('Analythical solve n >> 10')

figure('Name', 'Numeric solve n >> 10') plot(1:N, psi(:,1)) title('Numeric solve n >> 10')



# Ex. 5 (Advanced) En ~ n by numeric solve

In ex.5 we need to find out dependence of numeric En from n by plotting.

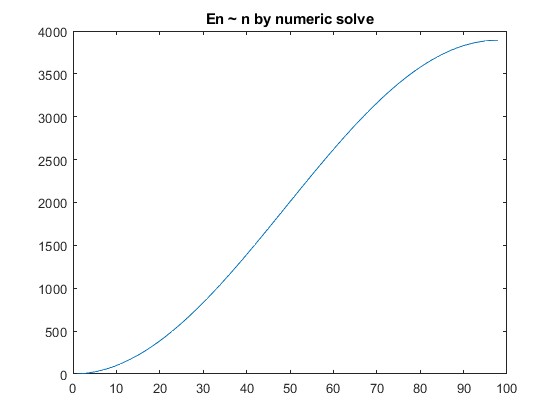
figure('Name', 'En ~ n by numeric solve') n = 1:98; h = 1;

t0 = hbar^2 / (2 \* 0.07 \* m0 \* h); En = 2 \* t0 \* (1 - cos(n \* pi \* h / 98)); plot(n, En./En(1)) title('En ~ n by numeric solve')

datetime(clock)

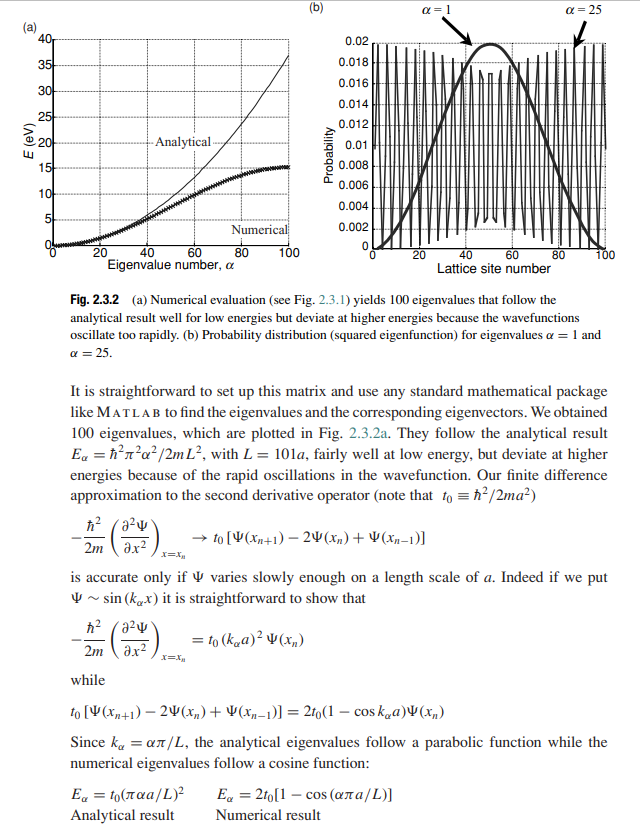
*ans =*

*datetime 09-Nov-2023 22:30:55*



*Published with MATLAB® R2023a*

Теория к 5му номеру



L05\_20231028

Цель: изучение численных методов на примере решения уравнения Шредингера для 2D потенциальной ямы

**Table of Contents**

........................................................................................................................................................ 1

Handle-functions ................................................................................................................................. 1 Ex.1 Analythical solve ......................................................................................................................... 1 Ex.1 Numeric solve ............................................................................................................................. 2

Ex.2 abs(psi)^2 (Analythical solve) ........................................................................................................ 3 Ex.2 abs(psi)^2 (Numeric solve) ............................................................................................................ 4 Ex.3 E~n An. and Num. solves ............................................................................................................. 5 Ex. 4 An. and Num. solves for n >> 10 .................................................................................................. 6 Ex. 5 (Advanced) En ~ n by numeric solve ............................................................................................. 8

clc, clear, close all load 'constants.mat'

# Handle-functions

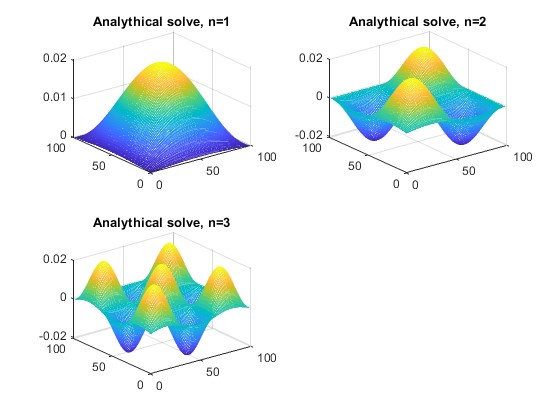
Energy = @ (n, L, meff) (hbar \* pi \* n / (L \* 1e-9))^2 / (2 \* meff \* m0) \*

J2eV \* 1e3;

# Ex.1 Analythical solve

In ex.1 we need to compare analythical and numeric solves for n = 1, 2, 3 by plotting.

figure('Name', 'Analythical solve') hold on L = 98; h = 1; for n = 1:3 for x = 1:L for y = 1:L Psin(x, y) = (2 / L) \* (sin(pi \* n \* x / L)) \* (sin(pi \* n \* y / L)); end end nexttile mesh(1:L, 1:L, Psin) title(strcat('Analythical solve, n= ', num2str(n))) end hold off



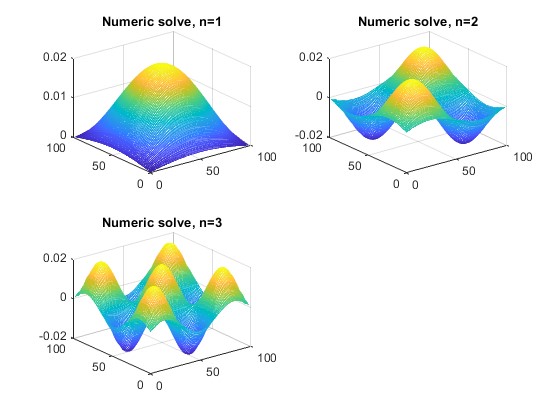
# Ex.1 Numeric solve

figure('Name', 'Numeric solve') N = 98;

I = eye(N, N); T = I; for i = 1:N for j = 1:N if i == j T(i, j) = 4; continue end

if i == j - 1 || i == j + 1 || i == j - 5 || i == j + 5 T(i, j) = -1; end end end [psi, D] = eig(T); psi = psi ./ sqrt(h);

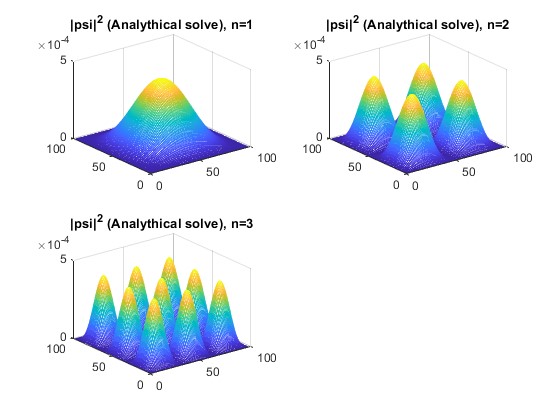
for n = 1:3 vals = psi(:, n); for x = 1:L for y = 1:L psi\_r(x, y) = vals(x) \* vals(y); end end nexttile mesh(1:L, 1:L, psi\_r) title(strcat('Numeric solve, n= ', num2str(n))) end



# Ex.2 abs(psi)^2 (Analythical solve)

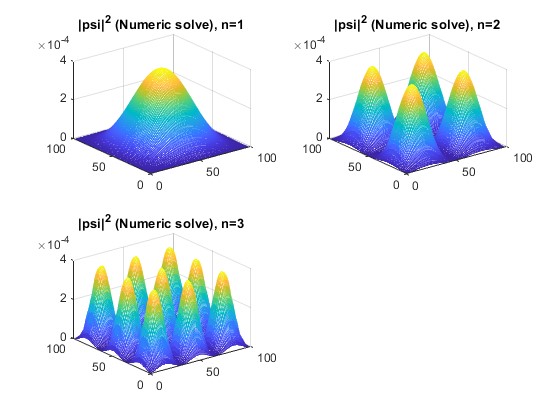
In ex.2 we need the same thing we had in ex.1 but for abs(psi)^2.

figure('Name', 'abs(psi)^2 (Analythical solve)') hold on for n = 1:3 for x = 1:L for y = 1:L Psin(x, y) = (2 / L) \* (sin(pi \* n \* x / L)) \* (sin(pi \* n \* y / L)); end end nexttile mesh(1:L, 1:L, abs(Psin).^2) title(strcat('|psi|^2 (Analythical solve), n= ', num2str(n))) end hold off



# Ex.2 abs(psi)^2 (Numeric solve)

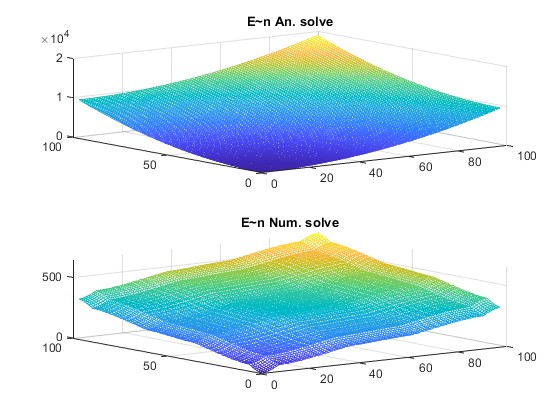
figure('Name', 'abs(psi)^2 (Numeric solve)') for n = 1:3 vals = psi(:, n); for x = 1:L for y = 1:L psi\_r(x, y) = abs(vals(x) \* vals(y))^2; end end nexttile mesh(1:L, 1:L, psi\_r) title(strcat('|psi|^2 (Numeric solve), n= ', num2str(n))) end



# Ex.3 E~n An. and Num. solves

In ex.3 we need to compare analythical and numeric connection between E and n by plotting. Also we need to try high n values (n = 10, 50, 90).

figure('Name', 'E~n Num. and An. solves') E = hbar^2 \* diag(D) / (h^2 \* 2 \* m0); for nx = 1:98 for ny = 1:98 Enum(nx, ny) = E(nx)/E(1) + E(ny)/E(1); Ean(nx, ny) = Energy(nx, 98, 0.07)/Energy(1, 98, 0.07) + ... Energy(ny, 98, 0.07)/Energy(1, 98, 0.07); end end hold on nexttile mesh(1:98, 1:98, Ean) title('E~n An. solve') nexttile mesh(1:98, 1:98, Enum) title('E~n Num. solve') hold off



# Ex. 4 An. and Num. solves for n >> 10

In ex.4 we need the same thing we had in ex.1 but for n >> 10 (for example, n = 50).

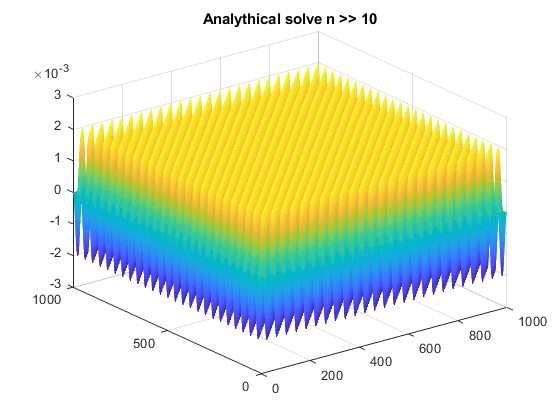
figure('Name', 'Analythical solve n >> 10') L = 998; h = 1; n = 50; for x = 1:L for y = 1:L Psin(x, y) = (2 / L) \* (sin(pi \* n \* x / L)) \* (sin(pi \* n \* y / L)); end end

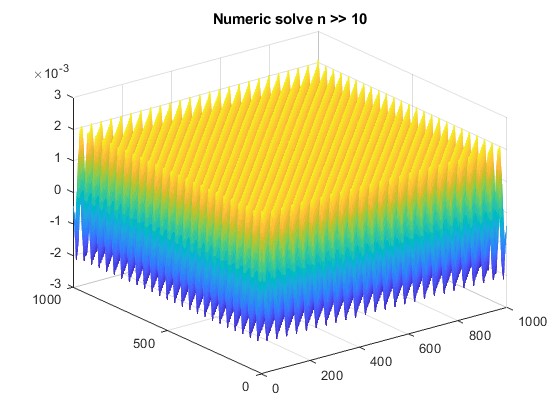
mesh(1:L, 1:L, Psin) title('Analythical solve n >> 10')

figure('Name', 'Numeric solve n >> 10') N = 998;

I = eye(N, N); T = I; for i = 1:N for j = 1:N if i == j T(i, j) = 4; continue end

if i == j - 1 || i == j + 1 || i == j - 5 || i == j + 5 T(i, j) = -1; end end end [psi, D] = eig(T); psi = psi ./ sqrt(h); vals = psi(:, n); for x = 1:N for y = 1:N psi\_r(x, y) = vals(x) \* vals(y); end end mesh(1:N, 1:N, psi\_r) title('Numeric solve n >> 10')





# Ex. 5 (Advanced) En ~ n by numeric solve

In ex.5 we need to confirm an equality of energies on levels like (21, 12), (13, 31), (23, 32).

figure('Name', 'En ~ n by numeric solve') En = [Enum(1, 1), Enum(2, 1), Enum(1, 2), Enum(1, 3), ... Enum(3, 1), Enum(2, 3), Enum(3, 2),]; plot(1:7, En)

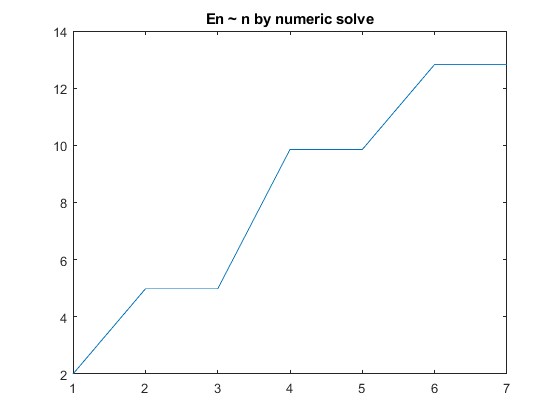
title('En ~ n by numeric solve')

datetime(clock)

*ans =*

*datetime*

*09-Nov-2023 23:58:46*



*Published with MATLAB® R2023a*

L06\_20231111\_FDTD

Цель: освоение метода конечных разностей с учётом временного фактора (FDTD), избавление опорного скрипта от циклов

% This is a 1D FDTD simulation of psi.

clear all, close all, clc

datetime(clock)

NN = 400; % Number of points in the problem space. hbar = 1.054e-34; % Plank's constant m0 = 9.1e-31; % Free space mass of an electron meff = 1.0; % Effective mass: Si is 1.08, Ge is 0.067, GaAs is 0.55 melec = meff\*m0; % Mass of an electron eV2J = 1.6e-19; % Energy conversion factors J2eV = 1/eV2J;

dx = .1e-9; % The cell size dt = 2e-17; % Time steps

ra = (0.5\*hbar/melec)\*(dt/dx^2); % ra must be < .15 DX = dx\*1e9; % Cell size in nm.

XX = (DX:DX:DX\*NN); % Length in nm for plotting % --- Specify the potential ---------------------

V=zeros(1,NN);

% --------------------------------------------------% Initialize a sine wave in a gaussian envelope

lambda = 50; % Pulse wavelength sigma = 50; % Pulse width nc = 150; % Starting position prl = zeros(1,NN); % The real part of the state variable pim = zeros(1,NN); % The imaginary part of the state variable ptot = 0.;

% n=2:NN-1;

prl = exp(-1.\*((n-nc)./sigma).^2).\*cos(2\*pi.\*(n-nc)./lambda) ; pim = exp(-1.\*((n-nc)./sigma).^2).\*sin(2\*pi.\*(n-nc)./lambda) ; ptot = sum(prl.^2 + pim.^2); % pnorm = sqrt(ptot); % Normalization constant

% Normalize and check ptot = 0.; prl = prl./pnorm; pim = pim./pnorm; ptot = ptot + sum(prl.^2 + pim.^2);

ptot % This should have the value 1

T = 0; n\_step = 1; count = 1; prl = [0, prl, 0]; pim = [0, pim, 0]; asked = false;

while count < 4 if asked == false n\_step = input('How many time steps -->'); asked = true; end if n\_step < 0 return end % -----------This is the core FDTD program -------------

T = T + 1; mid=2:NN-1; left = mid - 1; right = mid + 1;

prl(mid) = prl(mid) - ra.\*(pim(left) -2.\*pim(mid) + pim(right)) ... + (dt/hbar).\*V(mid).\*pim(mid);

pim(mid) = pim(mid) + ra.\*(prl(left) -2.\*prl(mid) + prl(right)) ... - (dt/hbar).\*V(mid).\*prl(mid);

% ------------------------------------------------------

% Calculate the expected values if n\_step == 1 PE = 0.;

psi = prl + i\*pim; % Write as a complex function PE = PE + sum(psi.\*transpose(psi').\*V);

psi\*psi' % This checks normalization PE = PE\*J2eV; % Potential energy

ke = 0. + j\*0.; lap\_p = psi(right) - 2\*psi(mid) + psi(left); ke = ke + sum(lap\_p\*psi(mid)'); KE = -J2eV\*((hbar/dx)^2/(2\*melec))\*real(ke); % Kinetic energy

subplot(3,1,count) plot(XX,prl,'k') hold on plot(XX,pim,'-.r') plot(XX,J2eV\*V,'--k') hold off

axis( [ 1 DX\*NN -.2 .3 ]) TT = text(5,.15,sprintf('%7.0f fs',T\*dt\*1e15)); set(TT,'fontsize',12)

TT = text(5,-.15,sprintf('KE = %5.3f eV',KE)); set(TT,'fontsize',12) TT = text(25,-.15,sprintf('PE = %5.3f eV',PE)); set(TT,'fontsize',12) TT = text(25,.13,sprintf('E\_t\_o\_t = %5.3f eV',KE+PE)); set(TT,'fontsize',12) xlabel('nm') set(gca,'fontsize',12) T

count = count + 1; asked = false;

end n\_step = n\_step - 1; end

*ans =*

*datetime 24-Nov-2023 20:34:42*

*ptot = 1.0000*

Error using input Cannot call INPUT from EVALC.

Error in L08\_20231111\_FDTD (line 61) n\_step = input('How many time steps -->');

*Published with MATLAB® R2023a*

0.2

0 fs

KE = 0.062 eV

PE = 0.000 eV

E

t

o

t

= 0.062 eV

0

-0.2

5 10 15 20 25 30 35 40

nm

0.2

34 fs

KE = 0.062 eV

PE = 0.000 eV

E

t

o

t

= 0.062 eV

0

-0.2

5 10 15 20 25 30 35 40

nm

0.2

68 fs

KE = 0.062 eV

PE = 0.000 eV

E

t

o

t

= 0.062 eV

0

-0.2

5 10 15 20 25 30 35 40

nm

L07\_20231125\_MOSFET

Цель: изучение работы MOSFET транзистора как системы потенциальных ям

% This is a 1D FDTD simulation of psi.

clear all, close all, clc

datetime(clock)

NN = 250; % Number of points in the problem space. hbar = 1.054e-34; % Plank's constant m0 = 9.1e-31; % Free space mass of an electron meff = 1.0; % Effective mass: Si is 1.08, Ge is 0.067, GaAs is 0.55 melec = meff\*m0; % Mass of an electron eV2J = 1.6e-19; % Energy conversion factors J2eV = 1/eV2J;

dx = 0.2e-9; % The cell size dt = 0.1e-15; % Time steps

ra = (0.5\*hbar/melec)\*(dt/dx^2); % ra must be < .15 DX = dx\*1e9; % Cell size in nm.

XX = (DX:DX:DX\*NN); % Length in nm for plotting % --- Specify the potential ---------------------

V=zeros(1,NN);

V(99:101) = 0.2 \* eV2J;

V(149:151) = 0.2 \* eV2J;

%V(102:148) = -3e-3 \* eV2J;

% --------------------------------------------------% Initialize a sine wave in a gaussian envelope

lambda = 200; % Pulse wavelength sigma = 100; % Pulse width nc = 100; % Starting position prl = zeros(1,NN); % The real part of the state variable pim = zeros(1,NN); % The imaginary part of the state variable ptot = 0.;

% %%n=2:NN-1; prl(2:0.4\*NN-1) = sin(2\*pi\*(2:0.4\*NN-1)/lambda); %pim = exp(-1.\*((n-nc)./sigma).^2).\*sin(2\*pi.\*(n-nc)./lambda) ; ptot = sum(prl.^2 + pim.^2); % pnorm = sqrt(ptot); % Normalization constant

% Normalize and check ptot = 0.; prl = prl./pnorm; pim = pim./pnorm; ptot = ptot + sum(prl.^2 + pim.^2);

ptot % This should have the value 1

T = 0; n\_step = 1;

count = 1; %prl = [0, prl, 0]; %pim = [0, pim, 0]; asked = false;

while count < 4 if asked == false n\_step = input('How many time steps -->'); asked = true; end if n\_step < 0 return end % -----------This is the core FDTD program -------------

T = T + 1; mid=2:NN-1; left = mid - 1; right = mid + 1;

prl(mid) = prl(mid) - ra.\*(pim(left) -2.\*pim(mid) + pim(right)) ... + (dt/hbar).\*V(mid).\*pim(mid);

pim(mid) = pim(mid) + ra.\*(prl(left) -2.\*prl(mid) + prl(right)) ... - (dt/hbar).\*V(mid).\*prl(mid); % ------------------------------------------------------

% Calculate the expected values if n\_step == 1 PE = 0.; psi = prl + 1i\*pim; % Write as a complex function %psi = psi(2:251);

%PE = PE + sum(psi.\*transpose(psi').\*V); %psi = [0,psi,0];

psi\*psi' % This checks normalization PE = PE\*J2eV; % Potential energy

ke = 0. + j\*0.; lap\_p = psi(right) - 2\*psi(mid) + psi(left); ke = ke + sum(lap\_p\*psi(mid)'); KE = -J2eV\*((hbar/dx)^2/(2\*melec))\*real(ke); % Kinetic energy

subplot(3,1,count) plot(XX,prl,'k') hold on plot(XX,pim,'-.r') plot(XX,J2eV\*V,'--k') hold off axis( [ 1 DX\*NN -.2 .3 ]) TT = text(5,.15,sprintf('%7.2f ps',T\*dt\*1e12)); set(TT,'fontsize',12) TT = text(5,-.15,sprintf('KE = %5.2f meV',KE\*1e3)); set(TT,'fontsize',12) TT = text(25,-.15,sprintf('PE = %5.3f eV',PE)); set(TT,'fontsize',12) TT = text(25,.13,sprintf('E\_t\_o\_t = %5.3f eV',KE+PE)); set(TT,'fontsize',12) xlabel('nm') set(gca,'fontsize',12) T

count = count + 1; asked = false;

end n\_step = n\_step - 1; end

*ans =*

*datetime 08-Dec-2023 03:30:18*

*ptot = 1*

Error using input Cannot call INPUT from EVALC.

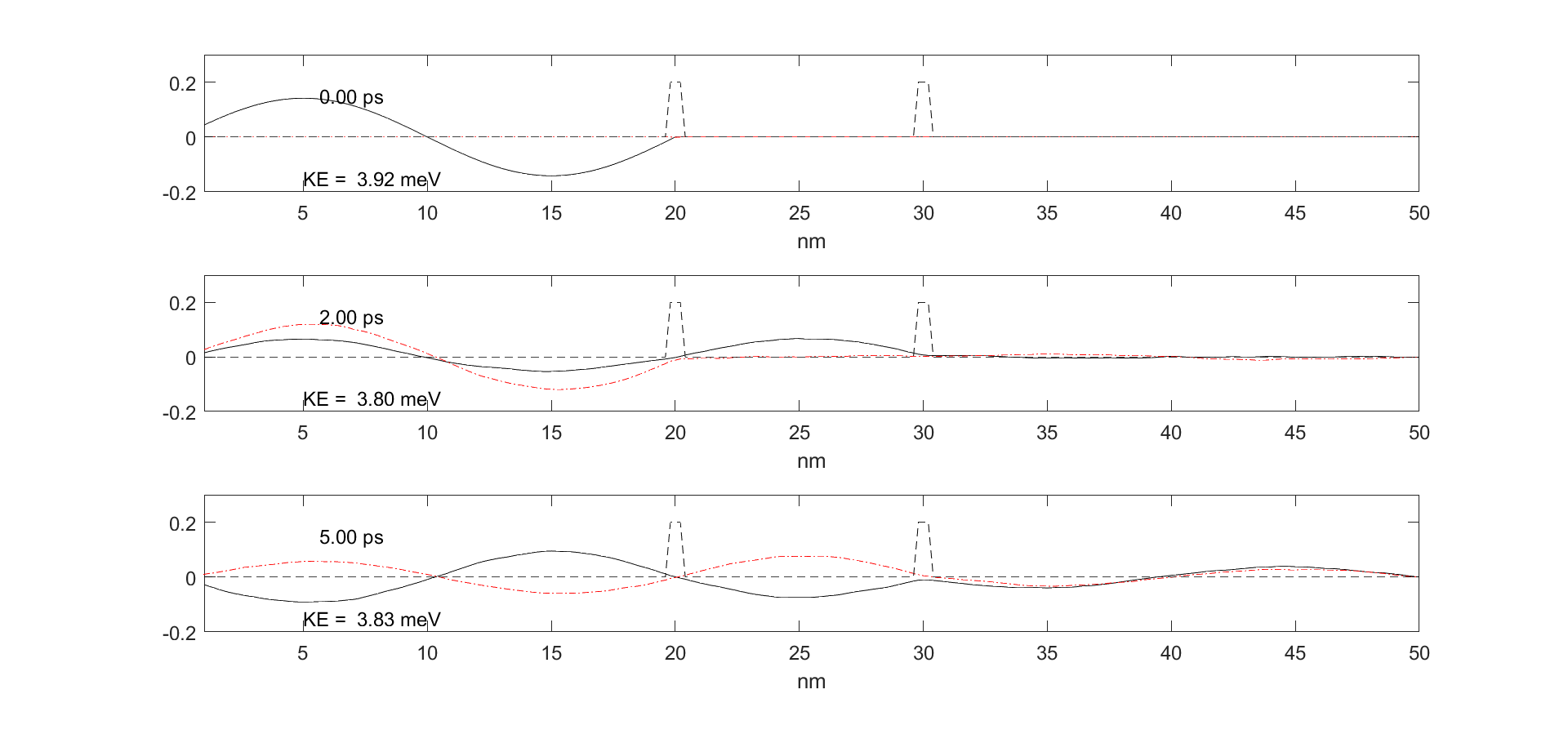
Error in MOSFET (line 63)

n\_step = input('How many time steps -->');

*Published with MATLAB® R2023a*

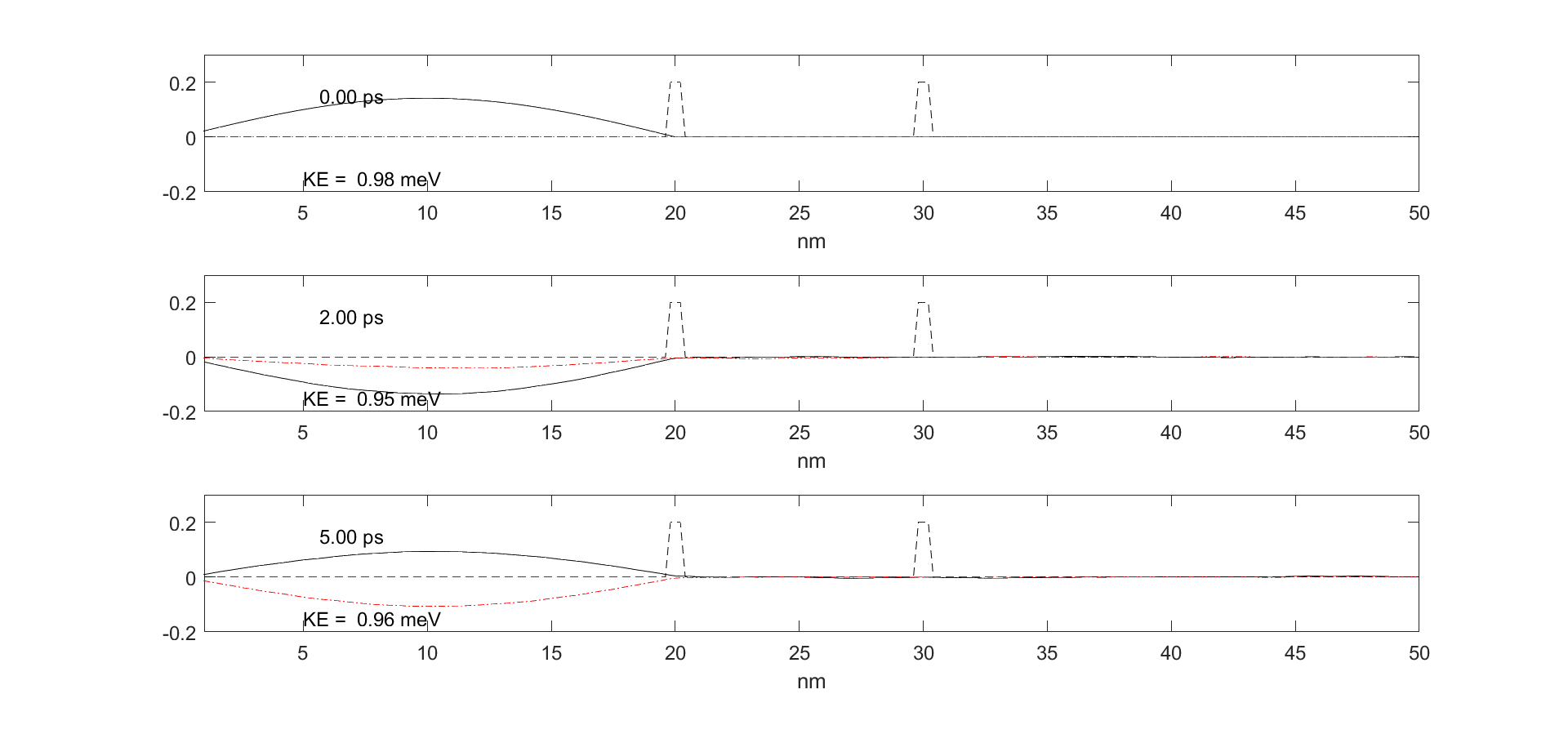
Задание 2

2е собственное состояние, лямбда = 100



Задание 3

Основное состояние, лямбда = 200

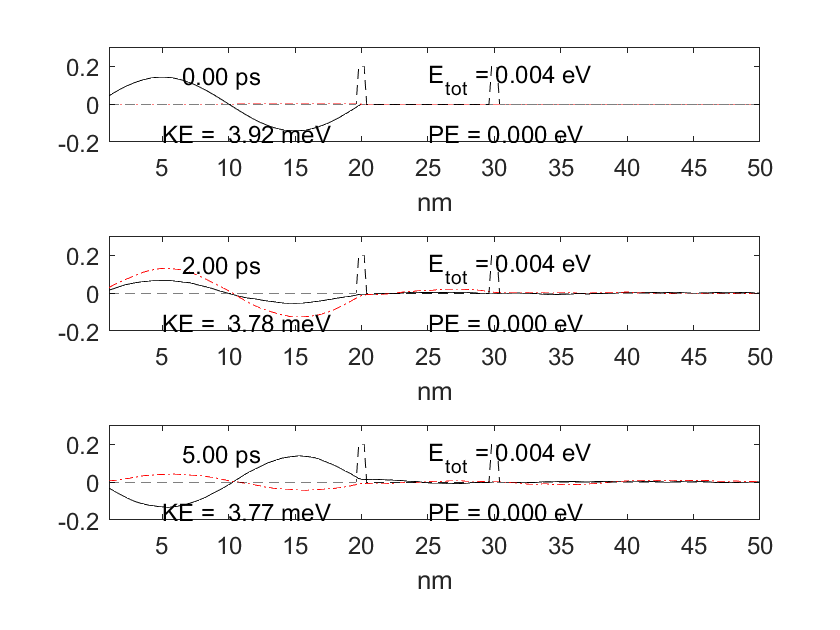


Задание 4

2е собственное состояние, лямбда = 100

Приложили потенциал 3 мэВ к средней яме

V(102:148) = -3e-3 \* eV2J;



Основное состояние, лямбда = 200

Приложили потенциал 3 мэВ к средней яме

V(102:148) = -3e-3 \* eV2J;

