**Mocanu Marian Valentin** Tema Laborator 1+2

2. Functia de o variabila

% definirea functiei

f = @(x) x.^4 - 6\*x.^2 + 4\*x + 12;

% generarea valorilor pentru x

x = linspace(-3, 3, 1000);

% calcularea valorilor functiei

y = f(x);

% reprezentarea grafica

figure;

plot(x, y, 'b', 'LineWidth', 2);

hold on;

xlabel('x');

ylabel('f(x)');

title('Identificarea minimelor locale și globale');

grid on;

% identificarea minimelor locale folosind gradientul functiei

syms x\_sym

f\_sym = x\_sym^4 - 6\*x\_sym^2 + 4\*x\_sym + 12;

df\_sym = diff(f\_sym); % Derivata funcției

% gasirea punctelor critice

crit\_points = double(solve(df\_sym == 0, x\_sym));

% calculul valorilor functiei in punctele critice

f\_crit = double(subs(f\_sym, x\_sym, crit\_points));

% afisarea minimelor pe grafic

plot(crit\_points, f\_crit, 'ro', 'MarkerSize', 8, 'MarkerFaceColor', 'r');

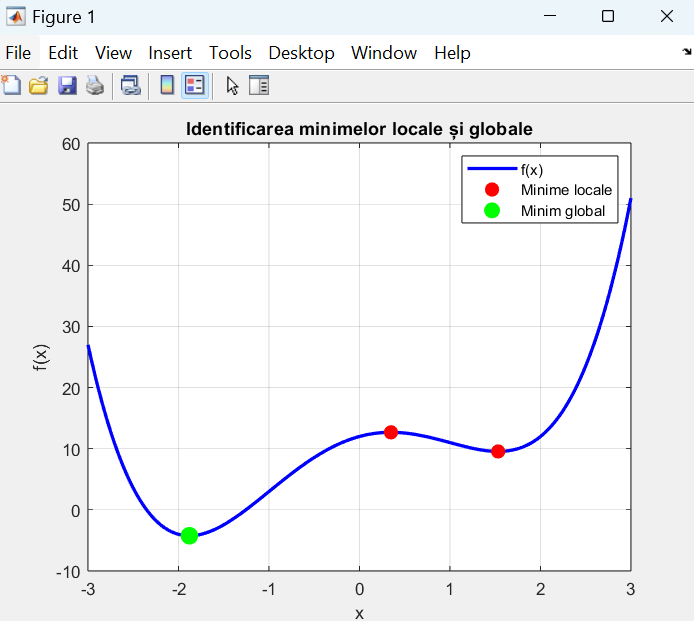
% determinarea minimului global intr-un interval

[x\_min, f\_min] = fminbnd(f, -3, 3);

plot(x\_min, f\_min, 'go', 'MarkerSize', 10, 'MarkerFaceColor', 'g');

legend('f(x)', 'Minime locale', 'Minim global');

hold off;



1. Functia de doua variabile

% definirea functiei de doua variabile

f = @(x, y) (1 - x).^2 + 100\*(y - x.^2).^2;

% definirea domeniului

x = linspace(-2, 2, 100);

y = linspace(-1, 3, 100);

[X, Y] = meshgrid(x, y);

Z = f(X, Y);

% reprezentarea 3D a functiei

figure;

surf(X, Y, Z, 'EdgeColor', 'none');

xlabel('x');

ylabel('y');

zlabel('f(x, y)');

title('Reprezentarea 3D a functiei Rosenbrock');

colorbar;

hold on;

% determinarea minimului global folosind optimizare neliniara

x0 = [0, 0];

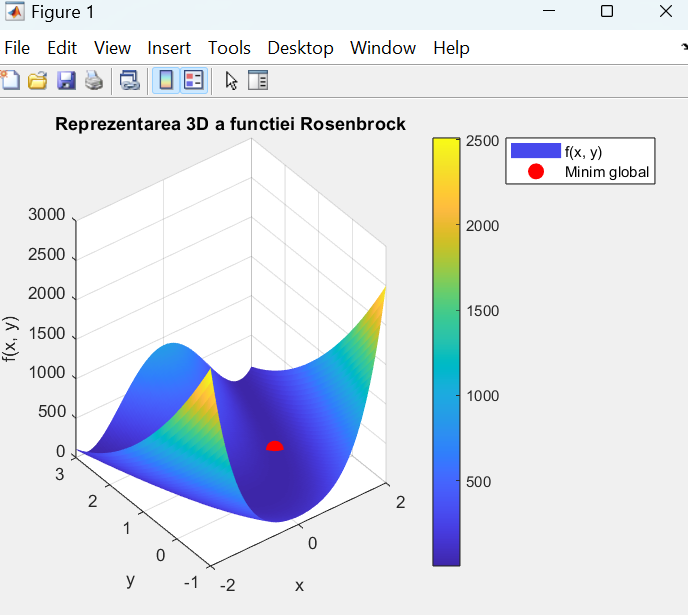
[x\_min, f\_min] = fminunc(@(v) f(v(1), v(2)), x0);

% marcarea minimului global pe grafic

plot3(x\_min(1), x\_min(2), f\_min, 'ro', 'MarkerSize', 10, 'MarkerFaceColor', 'r');

legend('f(x, y)', 'Minim global');

hold off;



2. Functia cu minime si maxime locale marcate

x = linspace(-2, 2, 100);

f = x.^4 - 2\*x.^2 + x;

% gasim minimele locale

minime\_locale = islocalmin(f);

x\_min = x(minime\_locale);

f\_min = f(minime\_locale);

% gasim maximele locale

maxime\_locale = islocalmax(f);

x\_max = x(maxime\_locale);

f\_max = f(maxime\_locale);

% cream graficul principal

figure

plot(x, f, 'b', 'LineWidth', 2)

hold on

% adaugam punctele minime cu cercuri rosii

plot(x\_min, f\_min, 'ro', 'MarkerSize', 8, 'LineWidth', 2)

% adaugam punctele maxime cu cercuri verzi

plot(x\_max, f\_max, 'go', 'MarkerSize', 8, 'LineWidth', 2)

grid on

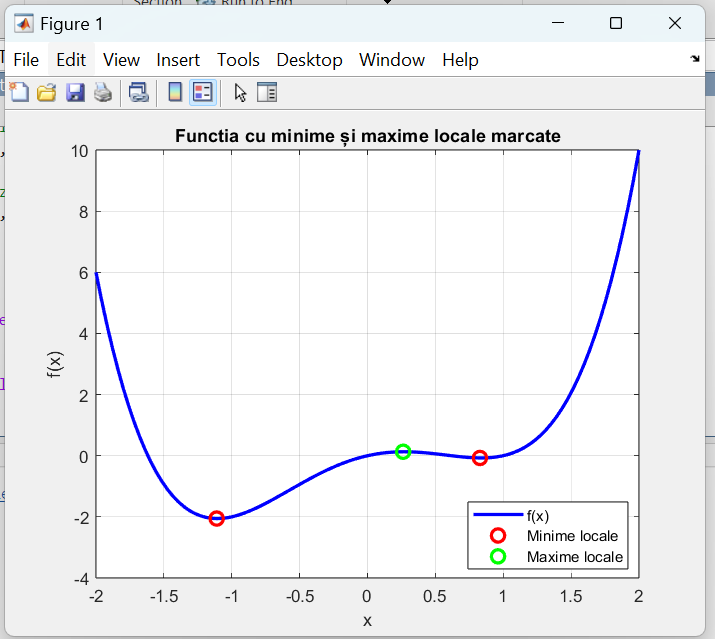
xlabel('x')

ylabel('f(x)')

title('Functia cu minime și maxime locale marcate')

legend('f(x)', 'Minime locale', 'Maxime locale', 'Location', 'best')

hold off



1. Functia de doua variabile cu multiple minime si maxime

[X, Y] = meshgrid(linspace(-5, 5, 50), linspace(-5, 5, 50));

F = sin(X) .\* cos(Y);

surf(X, Y, F)

xlabel('x')

ylabel('y')

zlabel('f(x, y)')

title('Functia de doua variabile cu multiple minime si maxime')

colormap jet

shading interp

% gasim punctele de minim și maxim folosind islocalmin și islocalmax

minime\_locale = islocalmin(F, 1) & islocalmin(F, 2);

maxime\_locale = islocalmax(F, 1) & islocalmax(F, 2);

% extragem coordonatele pentru minime si maxime

x\_min = X(minime\_locale);

y\_min = Y(minime\_locale);

f\_min = F(minime\_locale);

x\_max = X(maxime\_locale);

y\_max = Y(maxime\_locale);

f\_max = F(maxime\_locale);

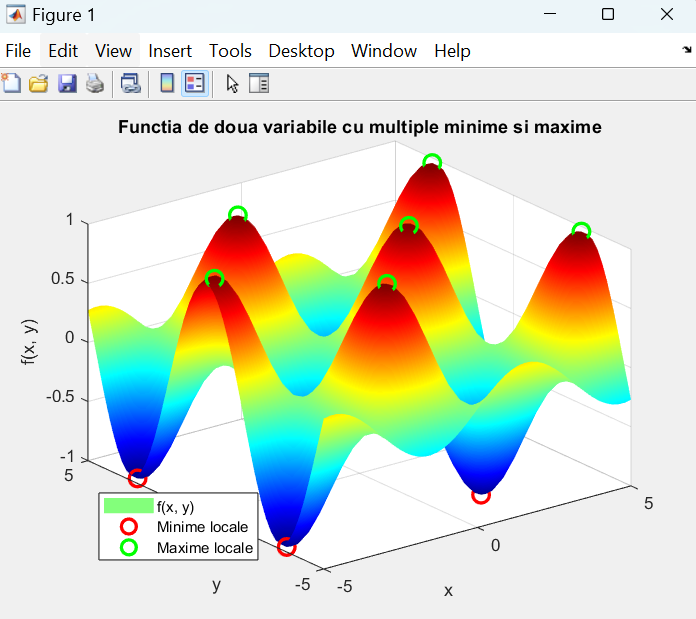
hold on

plot3(x\_min, y\_min, f\_min, 'ro', 'MarkerSize', 10, 'LineWidth', 2) % minime rosii

plot3(x\_max, y\_max, f\_max, 'go', 'MarkerSize', 10, 'LineWidth', 2) % maxime verzi

legend('f(x, y)', 'Minime locale', 'Maxime locale', 'Location', 'best')

hold off



1. Functia Rastrigin

[x, y] = meshgrid(linspace(-5, 5, 100), linspace(-5, 5, 100));

A = 10;

% calculam valorile functiei Rastrigin pentru fiecare punct (x, y)

rastrigin = A \* 2 + (x.^2 - A \* cos(2 \* pi \* x)) + (y.^2 - A \* cos(2 \* pi \* y));

% gasim minimele si maximele locale

minime\_locale = islocalmin(rastrigin, 1) & islocalmin(rastrigin, 2);

maxime\_locale = islocalmax(rastrigin, 1) & islocalmax(rastrigin, 2);

% extragem coordonatele pentru minime si maxime

x\_min = x(minime\_locale);

y\_min = y(minime\_locale);

f\_min = rastrigin(minime\_locale);

x\_max = x(maxime\_locale);

y\_max = y(maxime\_locale);

f\_max = rastrigin(maxime\_locale);

% afisam graficul 3D al functiei Rastrigin

figure;

surf(x, y, rastrigin, 'EdgeColor', 'none'); % Grafic 3D colorat

colormap jet; shading interp;

hold on;

% marcam minimele locale cu cercuri rosii

plot3(x\_min, y\_min, f\_min, 'ro', 'MarkerSize', 10, 'LineWidth', 2);

% marcam maximele locale cu cercuri verzi

plot3(x\_max, y\_max, f\_max, 'go', 'MarkerSize', 10, 'LineWidth', 2);

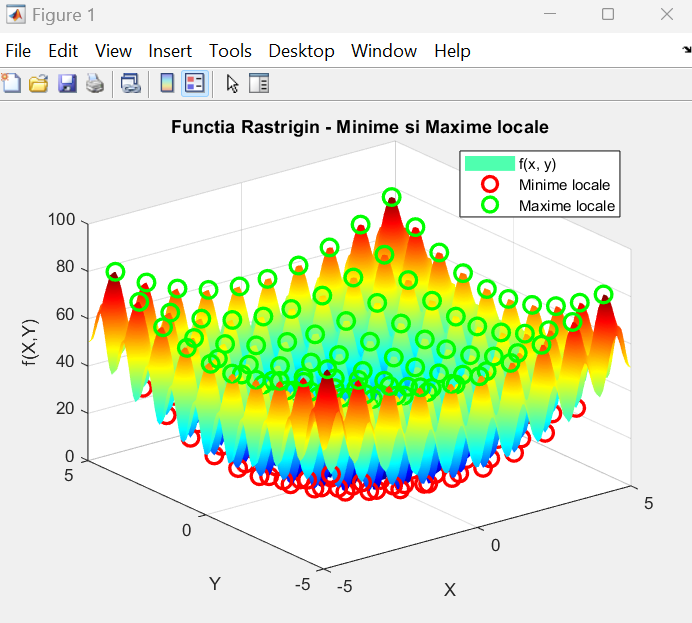
% setari pentru vizualizare

xlabel('X'); ylabel('Y'); zlabel('f(X,Y)');

title('Functia Rastrigin - Minime si Maxime locale');

legend('f(x, y)', 'Minime locale', 'Maxime locale', 'Location', 'best');

hold off;



1. Functiile Griewank si Ackley.

% definim domeniul

[X, Y] = meshgrid(linspace(-5,5,100));

% definim functia Griewank

Griewank = (X.^2 + Y.^2)/4000 - cos(X) .\* cos(Y/sqrt(2)) + 1;

% definim functia Ackley

A = 20; B = 0.2; C = 2\*pi;

Ackley = -A \* exp(-B \* sqrt((X.^2 + Y.^2)/2)) - exp((cos(C\*X) + cos(C\*Y))/2) + A + exp(1);

% afisam graficul pentru functia Griewank

figure;

surf(X, Y, Griewank);

title('Functia Griewank');

xlabel('X'); ylabel('Y'); zlabel('Valoare');

colormap autumn; shading interp;

% afisam graficul pentruu functia Ackley

figure;

surf(X, Y, Ackley);

title('Functia Ackley');

xlabel('X'); ylabel('Y'); zlabel('Valoare');

colormap autumn; shading interp;

% detectam minimele locale pentru functia Griewank

minimeGriewank = islocalmin(Griewank, 1) & islocalmin(Griewank, 2);

[maxX, maxY] = find(minimeGriewank);

% suprapunem minimele pe grafic

hold on;

scatter3(X(maxX, maxY), Y(maxX, maxY), Griewank(maxX, maxY), 50, 'r', 'filled');

% detectam minimele locale pentru functia Ackley

minimeAckley = islocalmin(Ackley, 1) & islocalmin(Ackley, 2);

[maxX, maxY] = find(minimeAckley);

% suprapunem minimele pe grafic

figure;

surf(X, Y, Ackley);

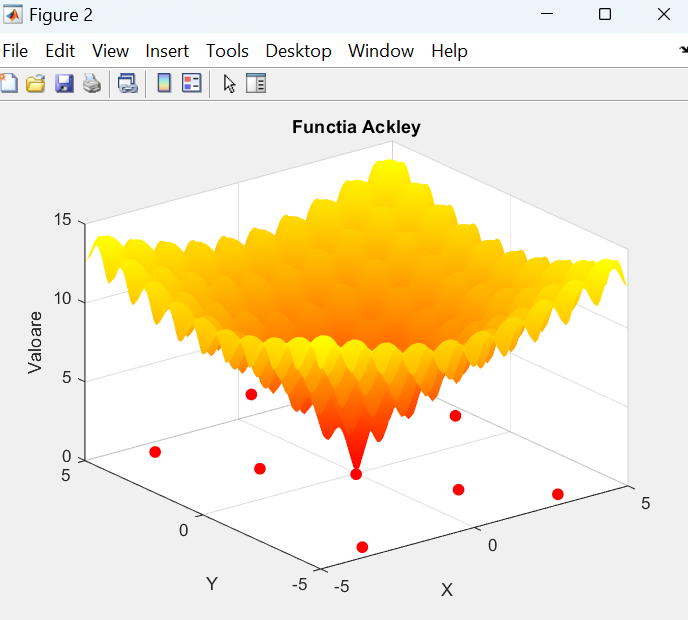
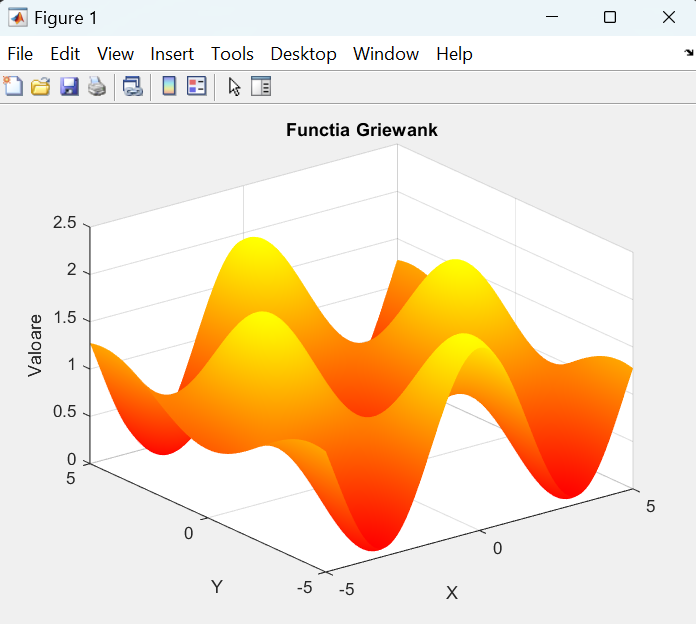
hold on;

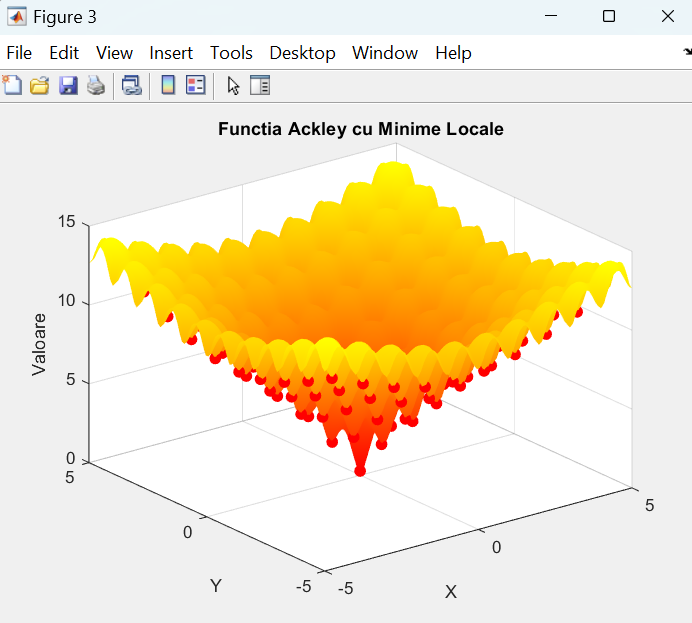
scatter3(X(maxX, maxY), Y(maxX, maxY), Ackley(maxX, maxY), 50, 'r', 'filled');

title('Functia Ackley cu Minime Locale');

xlabel('X'); ylabel('Y'); zlabel('Valoare');

colormap autumn; shading interp;







% definim domeniul [-5,5] pentru X si Y si functia f(x, y) = x^2 + y^2 + sin(x) + cos(y)

[X, Y] = meshgrid(linspace(-5,5,100));

F = X.^2 + Y.^2 + sin(X) + cos(Y);

% afisam grafic functia

figure;

surf(X, Y, F);

title('Functie cu un singur minim local');

xlabel('X'); ylabel('Y'); zlabel('f(X,Y)');

colormap spring; shading interp;

% evidentiem minimul local

[minVal, minIdx] = min(F(:)); % cautam minimul

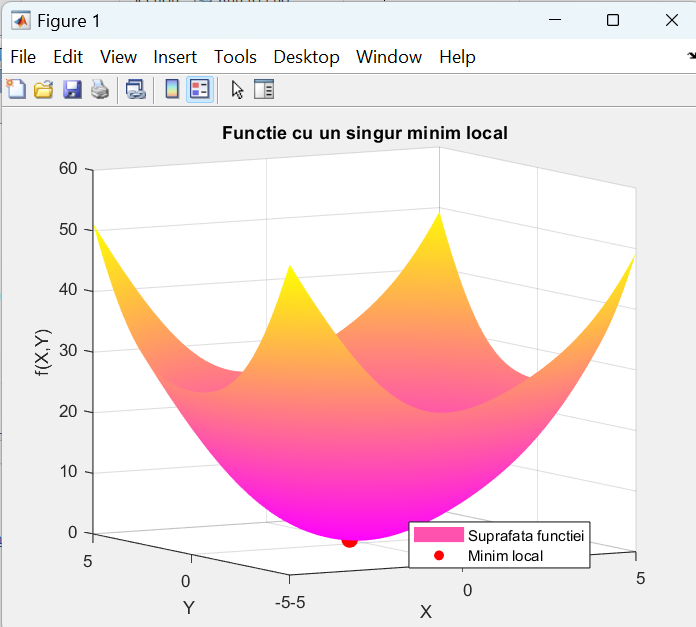
[minX, minY] = ind2sub(size(F), minIdx);

hold on;

scatter3(X(minX, minY), Y(minX, minY), F(minX, minY), 100, 'r', 'filled');

legend({'Suprafata functiei', 'Minim local'}, 'Location', 'best');

hold off;





syms x y

% --> functie de o variabila <--

f1 = x^4 - 3\*x^2 + 2;

df1 = diff(f1, x); % derivata primei functii

sol1 = double(solve(df1 == 0, x)); % punctele stationare convertite numeric

% determinam natura punctelor

D2f1 = diff(df1, x);

D2f1\_vals = double(subs(D2f1, x, sol1)); % evaluam derivata a doua

min\_points1 = sol1(D2f1\_vals > 0); % selectam punctele unde D2f1 este pozitiv

% --> functie de doua variabile <--

f2 = x^2 + y^2 + sin(x\*y);

grad\_f2 = gradient(f2, [x, y]); % gradientul

sol2 = solve(grad\_f2 == [0; 0], [x, y]); % punctele stationare

% calculam matricea Hessiana

H = hessian(f2, [x, y]);

H\_eval = double(subs(H, [x, y], [sol2.x, sol2.y]));

% verificam daca este minim local

is\_min = all(eig(H\_eval) > 0);

% realizam graficul pentru functia f2

[X, Y] = meshgrid(linspace(-5, 5, 100));

F2 = X.^2 + Y.^2 + sin(X.\*Y);

figure;

surf(X, Y, F2);

xlabel('X'); ylabel('Y'); zlabel('f(X,Y)');

title('Minimele functiei f(X,Y)');

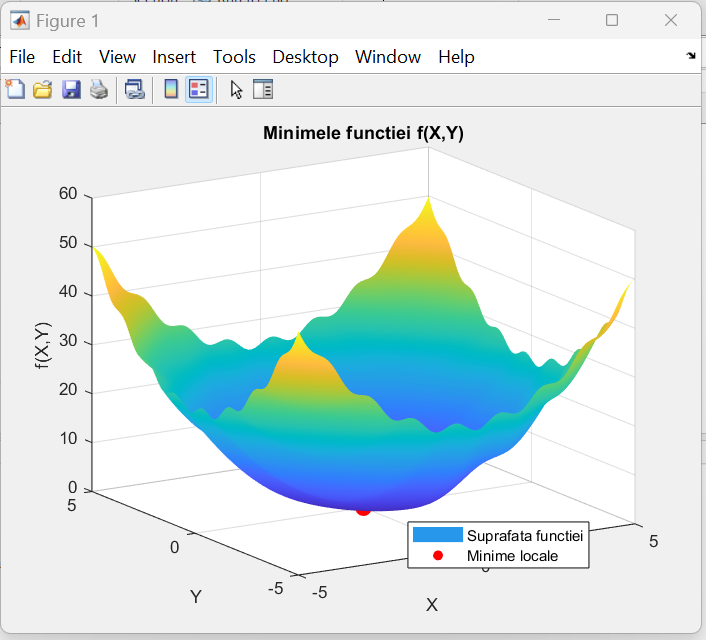
colormap parula; shading interp;

hold on;

scatter3(double(sol2.x), double(sol2.y), double(subs(f2, [x, y], [sol2.x, sol2.y])), 100, 'r', 'filled');

legend({'Suprafata functiei', 'Minime locale'}, 'Location', 'best');

hold off;





syms x y

% --> functie de o variabila <--

f1 = x^4 - 3\*x^2 + 2;

df1 = diff(f1, x); % derivata primei functii

sol1 = double(solve(df1 == 0, x)); % punctele stationare convertite numeric

% determinam natura punctelor

D2f1 = diff(df1, x);

D2f1\_vals = double(subs(D2f1, x, sol1)); % evaluam derivata a doua

min\_points1 = sol1(D2f1\_vals > 0); % selectam punctele unde D2f1 este pozitiv

% --> functie de doua variabile <--

f2 = x^2 + y^2 + sin(x\*y);

grad\_f2 = gradient(f2, [x, y]); % gradientul

sol2 = solve(grad\_f2 == [0; 0], [x, y]); % punctele stationare

% calculam matricea Hessiana

H = hessian(f2, [x, y]);

H\_eval = double(subs(H, [x, y], [sol2.x, sol2.y]));

% verificam daca este minim local

is\_min = all(eig(H\_eval) > 0);

% realizam graficul pentru functia f2

[X, Y] = meshgrid(linspace(-5, 5, 100));

F2 = X.^2 + Y.^2 + sin(X.\*Y);

figure;

surf(X, Y, F2);

xlabel('X'); ylabel('Y'); zlabel('f(X,Y)');

title('Minimele functiei f(X,Y)');

colormap parula; shading interp;

hold on;

scatter3(double(sol2.x), double(sol2.y), double(subs(f2, [x, y], [sol2.x, sol2.y])), 100, 'r', 'filled');

legend({'Suprafata functiei', 'Minime locale'}, 'Location', 'best');

hold off;

% --> calculul derivatei intr-o directie <--

v = [1; 1]; % directia dorita

unit\_v = v / norm(v); % normalizam vectorul

dir\_deriv = dot(grad\_f2, unit\_v); % derivata directionala

dir\_deriv\_vals = double(subs(dir\_deriv, {x, y}, {sol2.x, sol2.y}));

% afisam rezultatele

fprintf('Derivata directionala in punctele stationare:\n');

for i = 1:length(sol2.x)

fprintf('In punctul (%.3f, %.3f): %.3f\n', double(sol2.x(i)), double(sol2.y(i)), dir\_deriv\_vals(i));

end

% vizualizare a gradientului

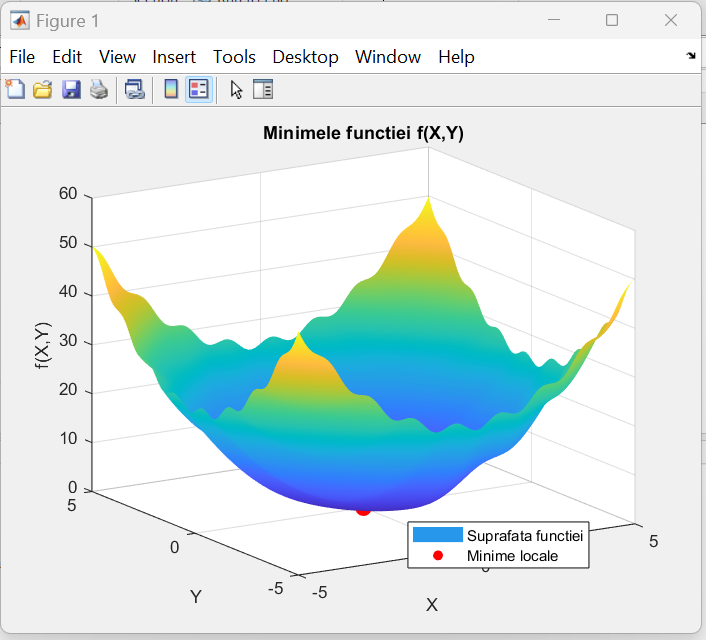
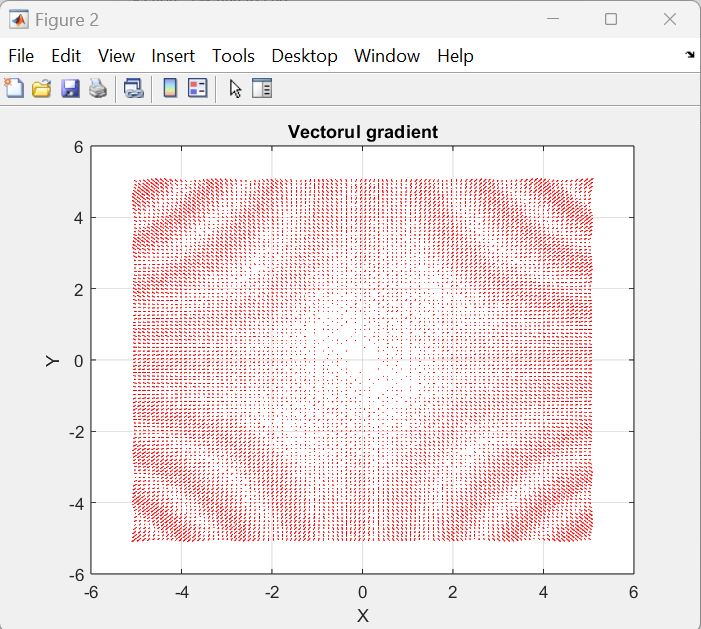
figure;

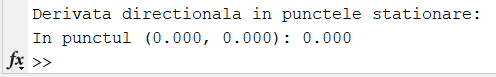
quiver(X, Y, double(subs(grad\_f2(1), {x, y}, {X, Y})), double(subs(grad\_f2(2), {x, y}, {X, Y})), 'r');

xlabel('X'); ylabel('Y');

title('Vectorul gradient');

grid on;





syms x y

% --> functie de o variabila <--

f1 = x^4 - 3\*x^2 + 2;

df1 = diff(f1, x); % derivata primei functii

sol1 = double(solve(df1 == 0, x)); % punctele stationare convertite numeric

% determinam natura punctelor

D2f1 = diff(df1, x);

D2f1\_vals = double(subs(D2f1, x, sol1)); % evaluam derivata a doua

min\_points1 = sol1(D2f1\_vals > 0); % selectam punctele unde D2f1 este pozitiv

% --> functie de doua variabile <--

f2 = x^2 + y^2 + sin(x\*y);

grad\_f2 = gradient(f2, [x, y]); % gradientul

sol2 = solve(grad\_f2 == [0; 0], [x, y]); % punctele stationare

% calculam matricea Hessiana

H = hessian(f2, [x, y]);

H\_eval = double(subs(H, [x, y], [sol2.x, sol2.y]));

% verificam daca este minim local

is\_min = all(eig(H\_eval) > 0);

% realizam graficul pentru functia f2

[X, Y] = meshgrid(linspace(-5, 5, 100));

F2 = X.^2 + Y.^2 + sin(X.\*Y);

figure;

surf(X, Y, F2);

xlabel('X'); ylabel('Y'); zlabel('f(X,Y)');

title('Minimele functiei f(X,Y)');

colormap parula; shading interp;

hold on;

scatter3(double(sol2.x), double(sol2.y), double(subs(f2, [x, y], [sol2.x, sol2.y])), 100, 'r', 'filled');

legend({'Suprafata functiei', 'Minime locale'}, 'Location', 'best');

hold off;

% --> calculul derivatei intr-o directie <--

v = [1; 1]; % directia dorita

unit\_v = v / norm(v); % normalizam vectorul

dir\_deriv = dot(grad\_f2, unit\_v); % derivata directionala

dir\_deriv\_vals = double(subs(dir\_deriv, {x, y}, {sol2.x, sol2.y}));

% afisam rezultatele

fprintf('Derivata directionala in punctele stationare:\n');

for i = 1:length(sol2.x)

fprintf('In punctul (%.3f, %.3f): %.3f\n', double(sol2.x(i)), double(sol2.y(i)), dir\_deriv\_vals(i));

end

% vizualizare a gradientului

figure;

quiver(X, Y, double(subs(grad\_f2(1), {x, y}, {X, Y})), double(subs(grad\_f2(2), {x, y}, {X, Y})), 'r');

xlabel('X'); ylabel('Y');

title('Vectorul gradient');

grid on;

% --> verificare FONC <--

punct\_test = [1; 1]; % Punct de test

val\_fonc = double(subs(grad\_f2, {x, y}, {punct\_test(1), punct\_test(2)}));

% afisam rezultatul

fprintf('Gradientul functiei in punctul (%.3f, %.3f) este: (%.3f, %.3f)\n', ...

punct\_test(1), punct\_test(2), val\_fonc(1), val\_fonc(2));

% reprezentam grafic gradientul in punctul testat

figure;

contour(X, Y, F2, 50);

hold on;

quiver(punct\_test(1), punct\_test(2), val\_fonc(1), val\_fonc(2), 'r', 'LineWidth', 2);

scatter(punct\_test(1), punct\_test(2), 100, 'b', 'filled');

xlabel('X'); ylabel('Y');

title('FONC - Vector gradient in punctul selectat');

grid on;

hold off;

