Optimizers everywhere -

Optimization in Financial Applications with MATLAB

Dr. Sarah Drewes
MathWorks Consulting Services

Optimization

Definition of *optimize* in English:

optimize

Oxford Dictionaries
Language matters

1 Make the best or most effective use of (a situation or resource):

Optimum (disambiguation)

From Wikipedia, the free encyclopedia

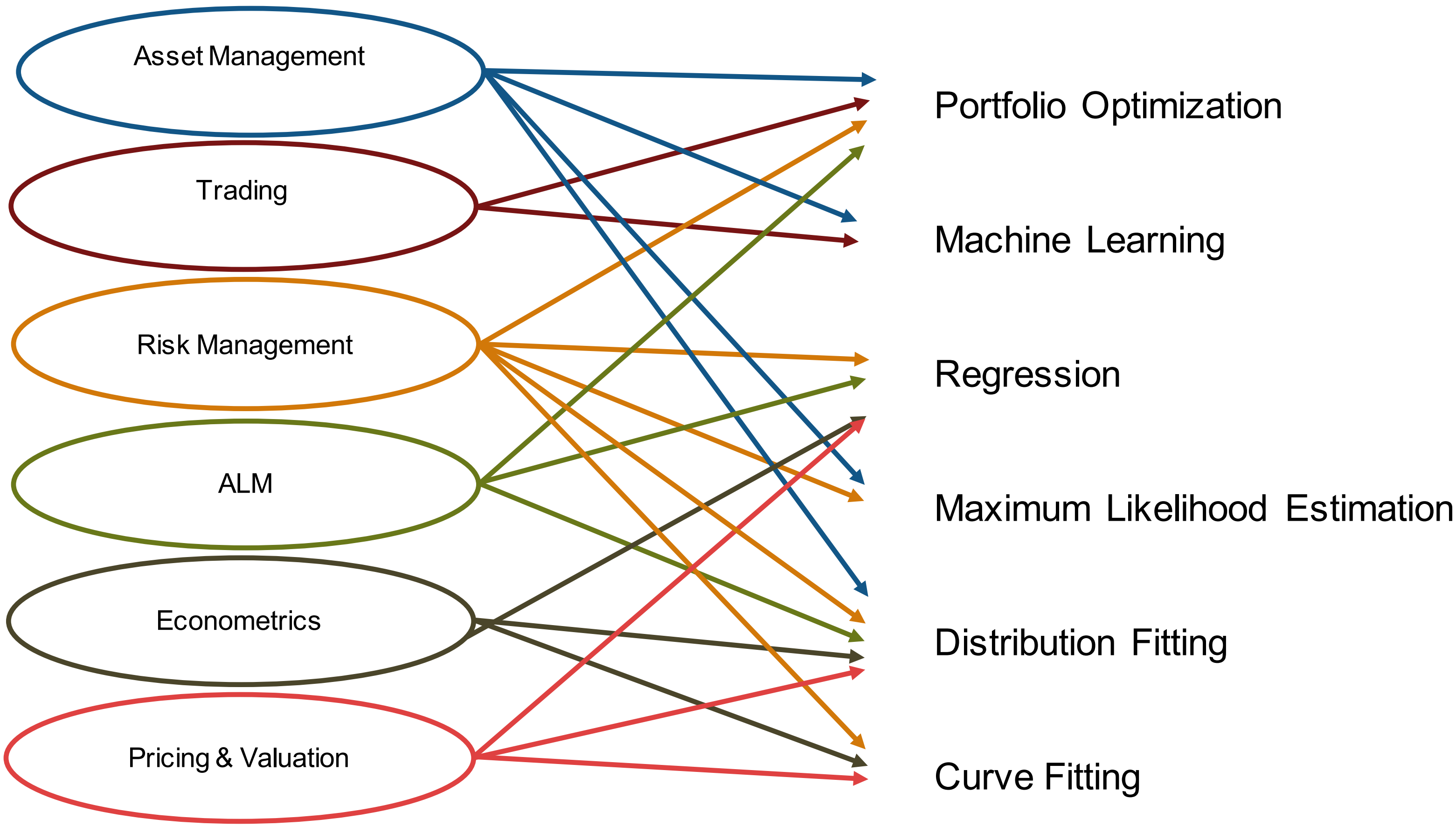
The **optimum** is the best or most favorable condition, or the greatest amount or degree possible under specific sets of comparable circumstances.

Optimization in Financial Applications with MATLAB

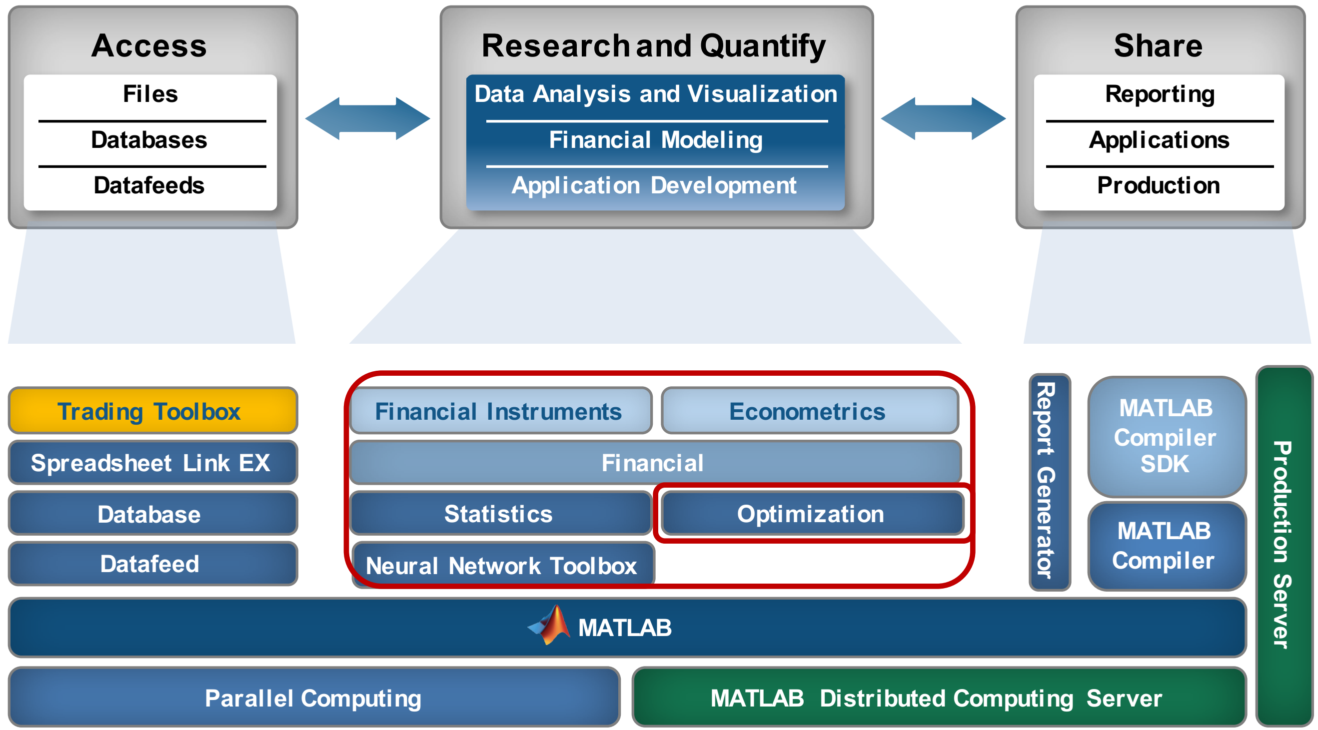
- Financial Optimization
- Optimization Methods
- Customized optimization models

Financial Optimization

Financial Applications and Optimization



MATLAB – The Financial Development Platform



Financial Optimization within MATLAB

Financial

– Mean-Variance Portfolio Optimization
– Conditional Value-at-Risk Portfolio Optimization
– Mean-Absolute Deviation Portfolios

Econometrics

– Time Series Regression Models
– Conditional Mean Variance Models
– Multivariate Models

Statistics

– Linear/ Nonlinear Regression
– Probability disribution fitting
– Machine Learning, e.g., SVM, NN,...

Neural Network

– Nonlinear Regression, Convolutional Neural Networks

Optimization

Optimization Methods

Optimization Problem

Objective Function

$$\min_{x} f(x)$$

Typically a linear or nonlinear function

Decision variables (can be discrete or integer)

Subject to Constraints

$$Ax \leq b \qquad c(x) \leq 0$$

$$A_{eq}x = b_{eq} \qquad c_{eq}(x) = 0$$

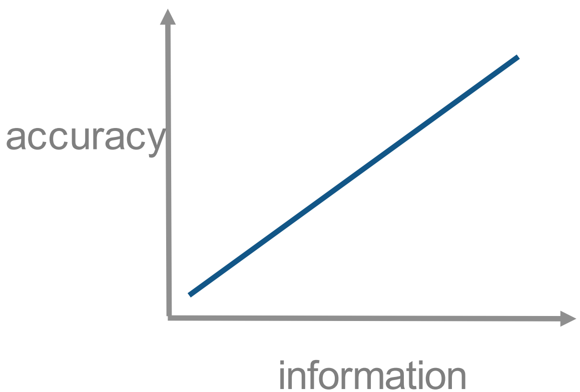
$$l \leq x \leq u$$

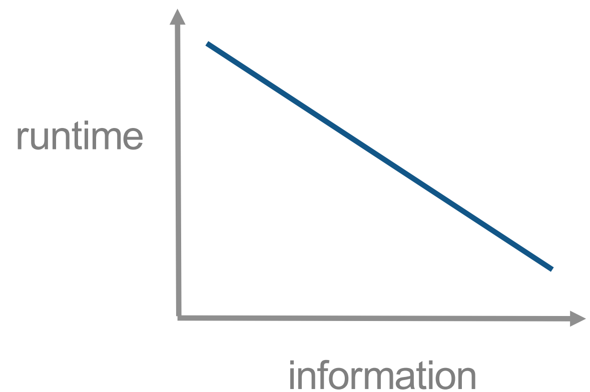
Linear constraints
- inequalities
- equalities
- bounds

Nonlinear constraints
- inequalities
- equalities

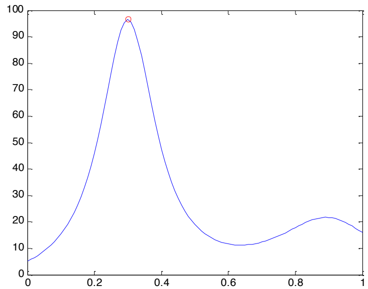
How to solve an optimization problem ?

What do you know about your optimization problem ?





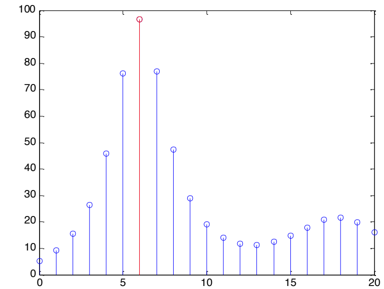
Variables & Constraints



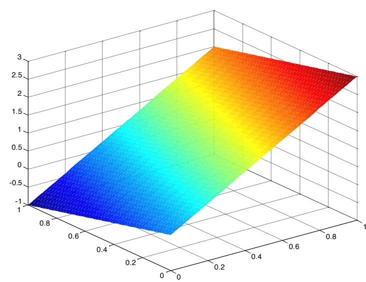
Variables

Continuous

Discrete



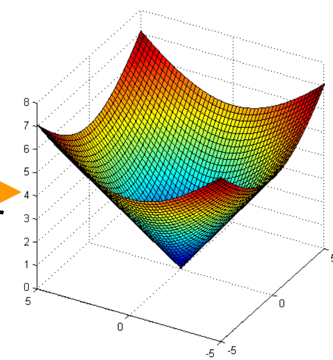
Integer Programming



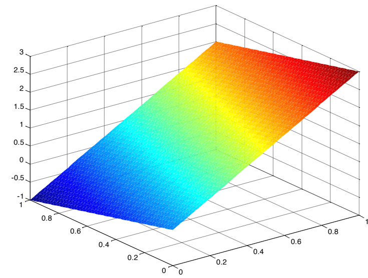
Constraints

Linear

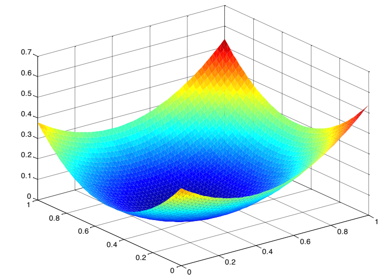
Nonlinear



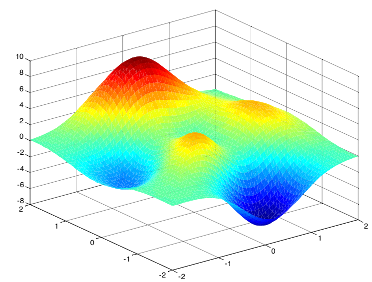
Objective Function



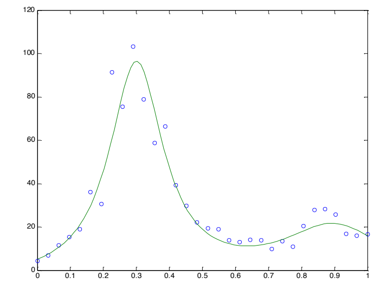
Linear $f(x) = A^{T}x$



Quadratic $f(x) = x^{T}Ax$



General $f(x)$

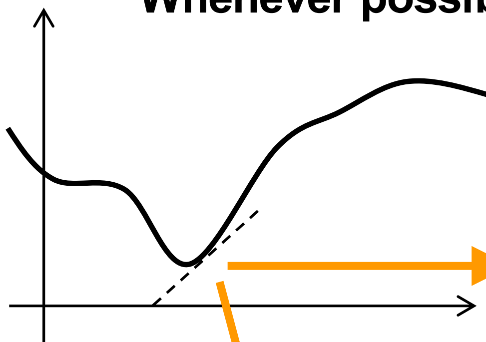


Least-squares/curve fitting

$f(c)=\Sigma[g(x_k;c) - y_k]^2$

Numerical optimization

Whenever possible, provide gradient/hessian information!

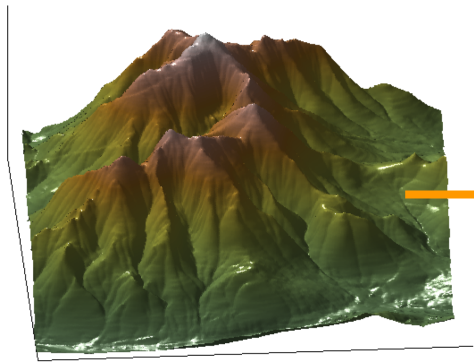


$$f'(x) \approx \frac{f(x+\Delta x) - f(x-\Delta x)}{2\Delta x}$$

```
function [f,df] = objective(x)
f = ...    % function value
df = ...   % gradient vector
```

- ✓ **Fewer function evaluations**
- ✓ **More accurate**

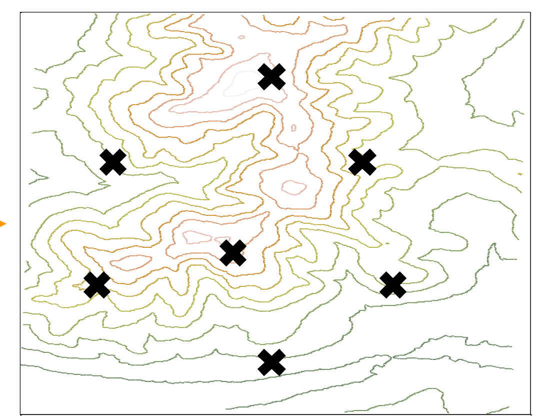
Derivative-Free Optimization



$\nabla f = ?$ → ~~`fmincon`~~

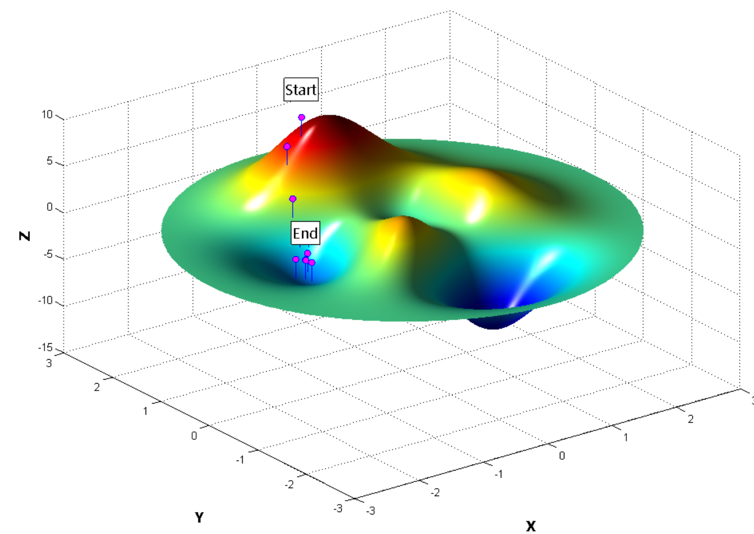
Repeatedly sample several points

Direct Search
Genetic Algorithm



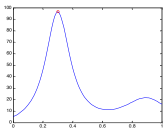
Approaches in MATLAB

- Local Optimization
  - Finds local minima/maxima
  - Uses supplied gradients or estimates them
  - Applicable for large scale problems with smooth objective function
  - Faster/fewer function evaluations
- Global Optimization
  - No gradient information required
  - Solve problems with non-smooth, discontinuous objective function

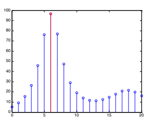


Solvers

Variables

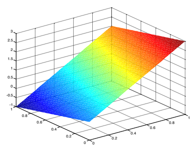


Continuous

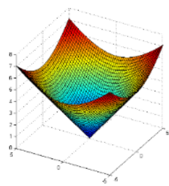


Discrete

Constraints

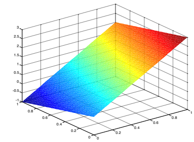


Linear

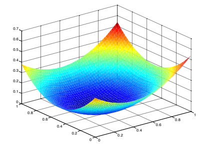


Nonlinear

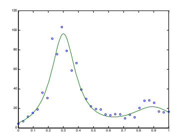
Objective function



Linear $f(x) = A^T x$

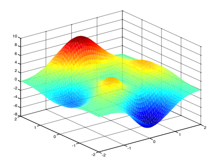


Quadratic $f(x) = x^T A x$



Least-squares

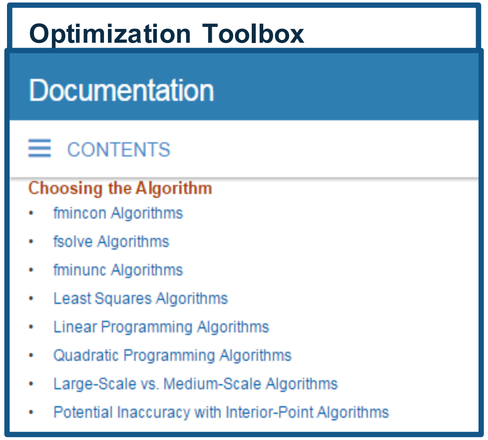
$f(c) = \Sigma[g(x_k;c) - y_k]^2$



General $f(x)$

Solver

`linprog`

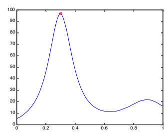


`fminunc`

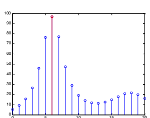
`fmincon`

Solvers

Variables

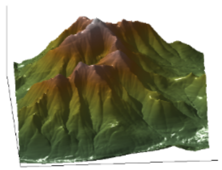


Continuous



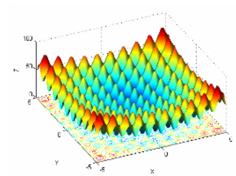
Discrete

Constraints



Non-smooth

Objective function



Nonlinear

Solver

Global Optimization Toolbox

Documentation

≡ CONTENTS

Choosing a Solver

Table for Choosing a Solver

There are seven Global Optimization Toolbox solvers:

- `ga` (genetic algorithm)
- `GlobalSearch`
- `MultiStart`
- `patternsearch`, also called direct search
- `particleswarm`
- `simulannealbnd` (simulated annealing)
- `gamultiobj`, which is not a minimizer; see Multiobjective Optimization

Customized Optimization Models

Supported Portfolio Optimization Models

Financial Toolbox

- Mean-Variance Portfolio Optimization
- Conditional Value-at-Risk Portfolio Optimization
- Mean-Absolute Deviation Portfolio Optimization

Customized Objective - Smart Beta

Customized Portfolio Optimization - Smart Beta

Smart Beta Example – Risk Parity:

Generate a portfolio where each asset's marginal contribution to risk is equal

- Marginal Contributions for N assets

$$MC_i(x) = x_i \times \frac{\sum_{j=1}^{N} x_j Cov(R_i, R_j)}{\sigma}$$

- Minimize distance between all contributions

$$f(x) = \sum_{i=1}^{N} \sum_{j \neq i} |MC_i(x) - MC_j(x)|$$

Customized Portfolio Optimization - Smart Beta

- Create marginal risk contributions

$$MC_i(x) = x_i \times \frac{\sum_{j=1}^{N} x_j Cov(R_i, R_j)}{\sigma}$$

- Create a distance matrix
- Minimize the total distance between all the contributions

$$f(x) = \sum_{i=1}^{N} \sum_{j \neq i} |MC_i(x) - MC_j(x)|$$

```
function out = riskCostFunction(wts , covMat)
% Cost function to get equal risk-contribution

% Build up all the marginal risk contributions
rMarg = zeros(size(wts));
for i = 1:length(rMarg)
    rMarg(i) = wts(i) * covMat(i,:)*wts';
end

% Build a distance matrix
[xx,yy] = meshgrid(rMarg);

% Compute total distance between entries
out = sum(sum(abs(xx-yy)));
```

Customized Portfolio Optimization - Smart Beta

Smart Beta Example – Risk Parity:

$$\min_{x} f(x) := \sum_{i=1}^{N} \sum_{j \neq i} |MC_i(x) - MC_j(x)|$$

s.t.

$$x^T \mathrm{e} = 1$$

$$\mathrm{lb} \leq \mathrm{x} \leq \mathrm{ub}$$

→ Nonlinear objective function with linear equality and bound constraints

Customized Portfolio Optimization - Smart Beta

- Solve with `fmincon` from MATLAB® Optimization Toolbox™

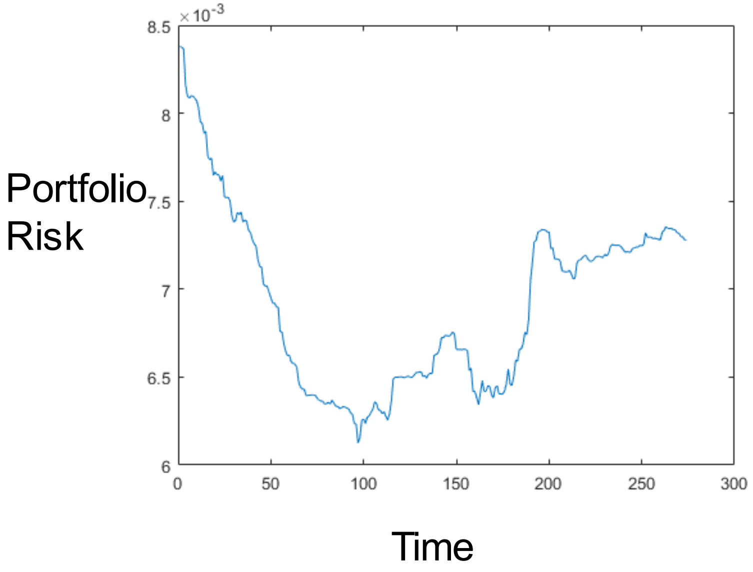
```
[fixWeights,fVal] = fmincon(@(x) riskCostFunction(x , covRet),...
                            x0 , [] , [] ,...
                            ones(1 , nAsset) , 1 ,...
                            LB ,...
                            UB , [] , o);
```

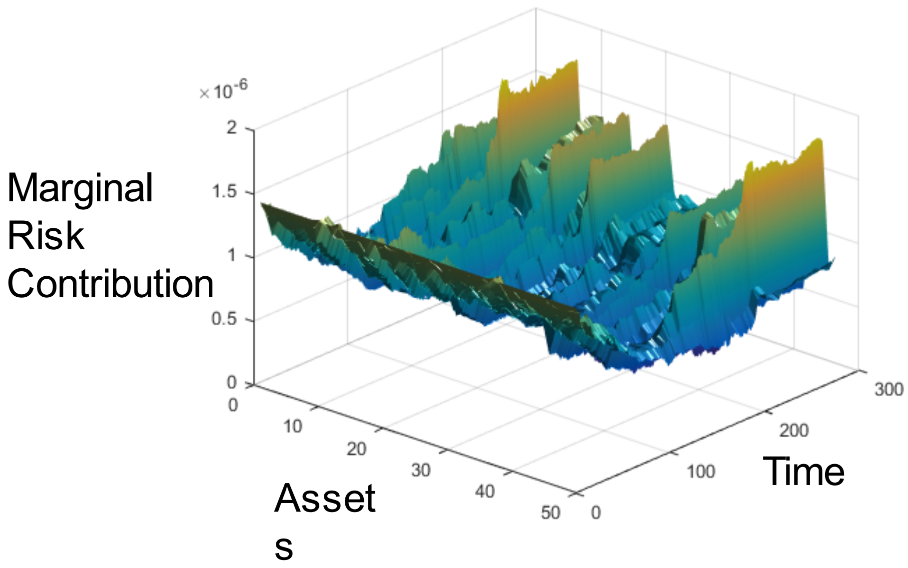
total distance between marginal risk contributions

budget constraint

bounds

Customized Portfolio Optimization - Smart Beta





Customized Constraints - Robust Constraints

Customized Portfolio Optimization - Robust Constraints

Mean-Variance Portfolio Optimization

$$\min_x x^T \boxed{C} x$$

$$s.t.\ \boxed{m}^T x \geq R$$

$$x^T e = 1$$

$$\mathrm{x} \geq 0$$

Covariance matrix

mean of asset returns

budget constraint

lower bound

Financial Toolbox

```
m = [ 0.05; 0.1; 0.12; 0.18 ];
C = [ 0.0064 0.00408 0.00192 0;
    0.00408 0.0289 0.0204 0.0119;
    0.00192 0.0204 0.0576 0.0336;
    0 0.0119 0.0336 0.1225 ];

p = Portfolio('assetmean', m, 'assetcovar', C, ...
'lowerbudget', 1, 'upperbudget', 1, 'lowerbound', 0);
```

Customized Portfolio Optimization - Robust Constraints

Additional constraints supported by **Financial Toolbox**

- Linear inequality constraints
- Linear equality constraints
- Group constraints
- Group ratio constraints
- Average turnover constraints
- One-way turnover constraints

Customized Portfolio Optimization-
Robust Constraints

Extend standard model by individual constraints, e.g. robust constraints

$$Prob(\boxed{\xi^T} x \geq \boxed{R}) \geq \boxed{p}$$

random vector of returns

probability level

return level

→Deterministic approximation (Chebychev's inequality)

$$\boxed{m}^T x - \sqrt{\frac{p}{1-p}} \sqrt{x^T \boxed{C} x} \geq R$$

mean vector of returns

covariance of returns

P. Bonami, M.A. Lejeune, ‚An Exact Solution Approach for Portfolio Optimization Problems Under Stochastic and Integer Constraints', Operations Research 2009, Vol. 57, Issue 3

Customized Portfolio Optimization - Robust Constraints

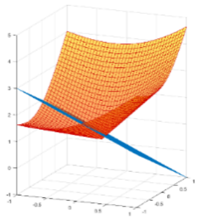
Mean-Variance Portfolio Optimization with robust constraint

$$\min_{x} f(x) := x^T C x$$

$$s.t \quad x^T e = 1 \quad \text{budget constraint}$$

$$x \geq 0 \quad \text{lower bound}$$

$$c(x) := R - m^T x + \sqrt{\frac{p}{1-p}} \sqrt{x^T C x} \leq 0 \quad \text{robust constraint}$$



→ Robust mean variance model is a nonlinear, convex NLP

Customized Portfolio Optimization- Robust Constraints

- Solve with `fmincon` from MATLAB® Optimization Toolbox™

```
[x,fval] =   fmincon(@(x)quadObj(x,C),...          %                 on
                  x0,[],[],...                     % start vector
                  ones(1,n),1,...                  %                 straint
                  zeros(n,1),inf(n,1),...          % lower and upper bounds
                  @(x)probConstr(x,p,m,C,R),...    % n               traints
                  options);
```

mean-variance risk proxy

budget constraint

robust constraint

Customized Portfolio Optimization-Robust Constraints

Provide gradients

```
options = optimoptions(@fmincon,...
    'SpecifyObjectiveGradient',true,...
    'SpecifyConstraintGradient',true, ...
```

```
function [f,gradf] = quadObj(x,C)

    f=x'*C*x;
    gradf = 2*C*x;

end
```

$f(x)$

$\nabla f(x)$

```
function [c,ceq, gradC,gradCeq ] = probConstr(x,p,m,C,R)

    denom = sqrt(x'*C*x);

    c = R - m'*x + p*denom;
    gradC =  -m + p/denom*C*x;

    ceq = [];
    gradCeq = [];

end
```

$c(x)$

$\nabla c(x)$

Customized Portfolio Optimization-
Robust Constraints

Provide Hessian

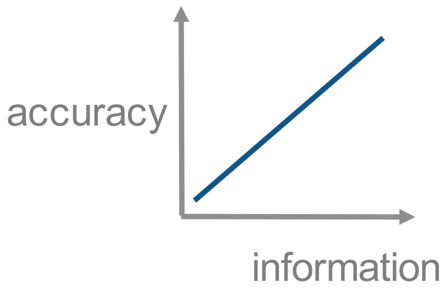
```
options = optimoptions(@fmincon,...
     'SpecifyObjectiveGradient',true,...
     'SpecifyConstraintGradient',true, ...
     'HessianFcn',@(x,lambda)getHessianRobustMV(x,lambda,C,p));
```

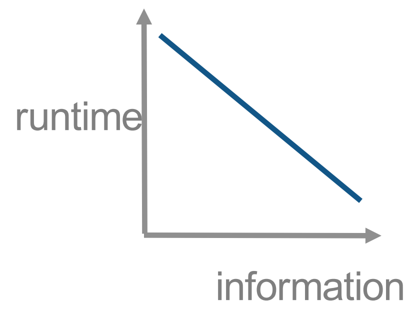
$$\nabla^2 f(x) + \sum \nabla^2 c_i(x) \cdot \lambda_i + \sum \nabla^2 ceq_j(x) \cdot \lambda_j$$

```
function H = getHessianRobustMV(x,lambda, C, p)

% This is the Hessian of
% objective function: quadObj
% constraint function: probConstr

sw = C*x;
dC = (p/sqrt(x'*C*x)).*(C-(sw*sw')./(x'*C*x));
dF = 2*C;
H =  dF + lambda.ineqnonlin(1)*dC;

end
```

Customized Portfolio Optimization - Robust Constraints

assets	result_no_gradients	time_no_gradients	result_with_gradients	time_with_gradients
50	'optimal'	0.1836	'optimal'	0.060784
100	'feasible'	0.56741	'optimal'	0.091847
500	'feasible'	8.5853	'optimal'	2.4931





Customized Portfolio Optimization - Robust Constraints

Without gradients

Iter	F-count	f(x)	Feasibility	optimality
0	51	6.098570e+01	2.400e+01	1.794e+02
1	102	1.881289e+01	1.757e+01	2.404e+02
2	153	8.850099e+00	1.172e+01	1.583e+02
3	204	2.926182e+00	5.126e+00	5.186e+01
4	255	1.599701e+00	2.437e+00	3.379e+00
5	306	1.269421e+00	5.720e-01	4.025e+00
6	358	1.249948e+00	3.610e-01	2.123e+00
7	409	1.279036e+00	1.805e-03	1.133e-01
8	460	1.042756e+00	9.026e-06	1.134e+00
9	511	5.778849e-01	6.602e-03	5.587e+00
10	562	4.839580e-01	9.742e-04	3.333e+00
11	613	4.824363e-01	9.517e-04	1.264e-01
12	664	3.615160e-01	4.711e-03	9.752e-01
13	715	3.328996e-01	1.925e-04	5.851e-02
14	766	3.204592e-01	6.739e-05	9.337e-02
15	817	3.150931e-01	1.709e-05	3.187e-02
16	868	3.113085e-01	4.433e-12	3.139e-02
17	919	3.108302e-01	1.244e-12	8.114e-03
18	970	3.108004e-01	8.873e-13	2.088e-04
19	1021	3.063753e-01	2.640e-13	5.799e-02
20	1072	3.031614e-01	2.625e-05	7.867e-03
21	1123	3.028625e-01	4.135e-06	4.636e-03
22	1174	3.028082e-01	5.297e-06	2.663e-03
23	1225	3.028131e-01	2.331e-15	3.900e-05
24	1276	3.028131e-01	2.554e-15	3.637e-06

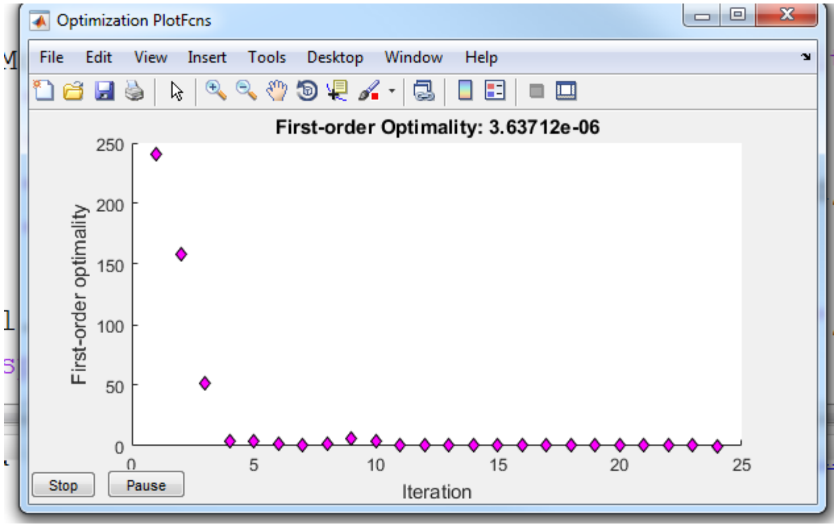
With gradients

Iter	F-count	f(x)	Feasibility	optimality
0	1	6.098570e+01	2.400e+01	1.794e+02
1	2	2.322288e+01	1.430e+01	1.269e+02
2	3	5.419761e+00	5.911e+00	2.018e+01
3	4	1.732759e+00	2.237e+00	5.623e+00
4	6	1.413489e+00	1.431e+00	2.503e+00
5	7	1.217778e+00	1.372e-01	1.418e-01
6	8	8.718262e-01	6.861e-04	1.491e+00
7	9	5.448938e-01	5.936e-03	3.887e+00
8	10	4.831605e-01	4.820e-04	4.325e+00
9	11	3.815757e-01	3.713e-03	1.552e+00
10	12	3.480601e-01	4.921e-04	2.120e+00
11	13	3.301950e-01	1.063e-10	1.102e+00
12	14	3.211171e-01	5.814e-11	2.573e-02
13	15	3.117667e-01	9.545e-05	3.841e-03
14	16	3.108083e-01	1.701e-12	2.069e-04
15	17	3.061063e-01	2.869e-13	6.978e-03
16	18	3.046152e-01	3.464e-12	9.367e-05
17	19	3.029598e-01	2.707e-06	1.094e-02
18	20	3.028149e-01	1.588e-14	1.571e-06

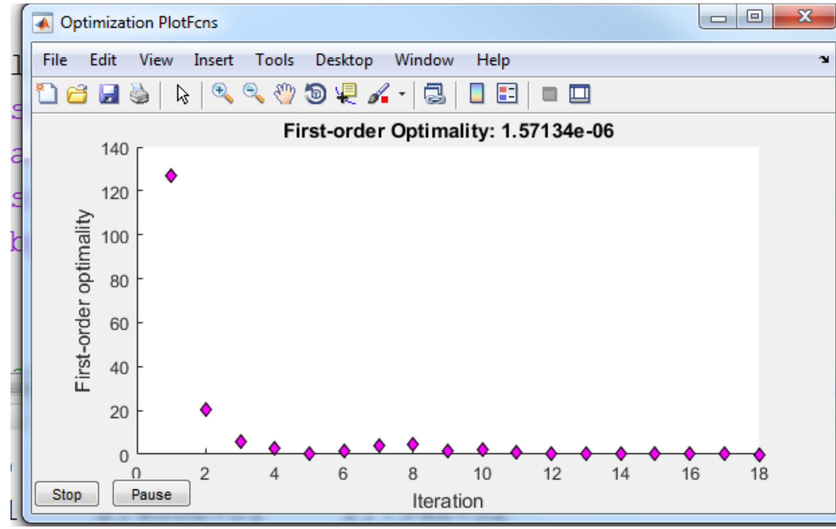
Customized Portfolio Optimization - Robust Constraints

```
'PlotFcn',@optimplotfirstorderopt
```

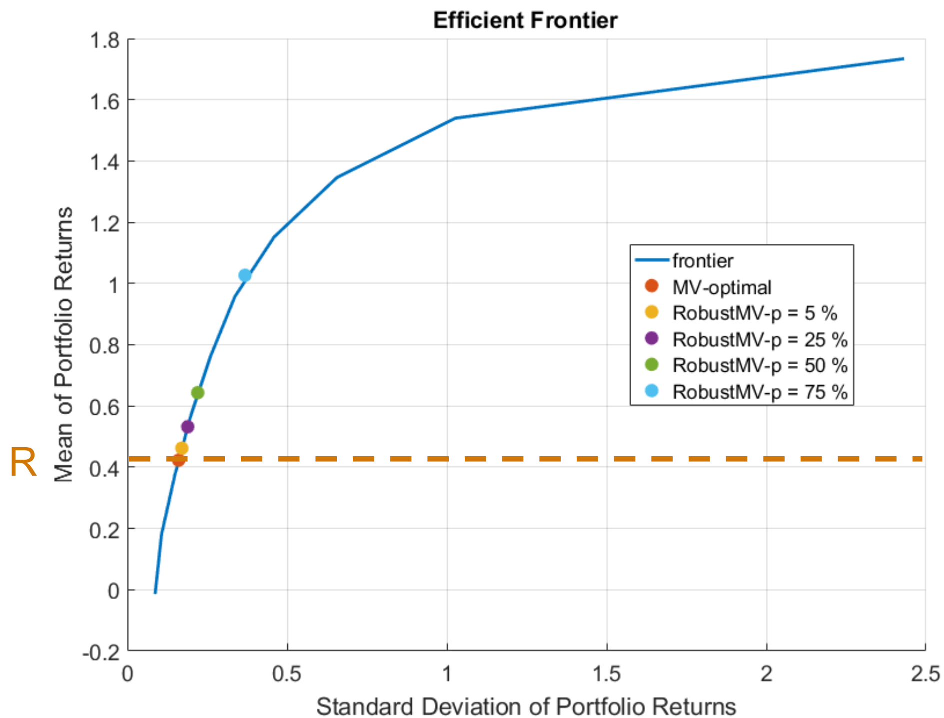
Without gradients



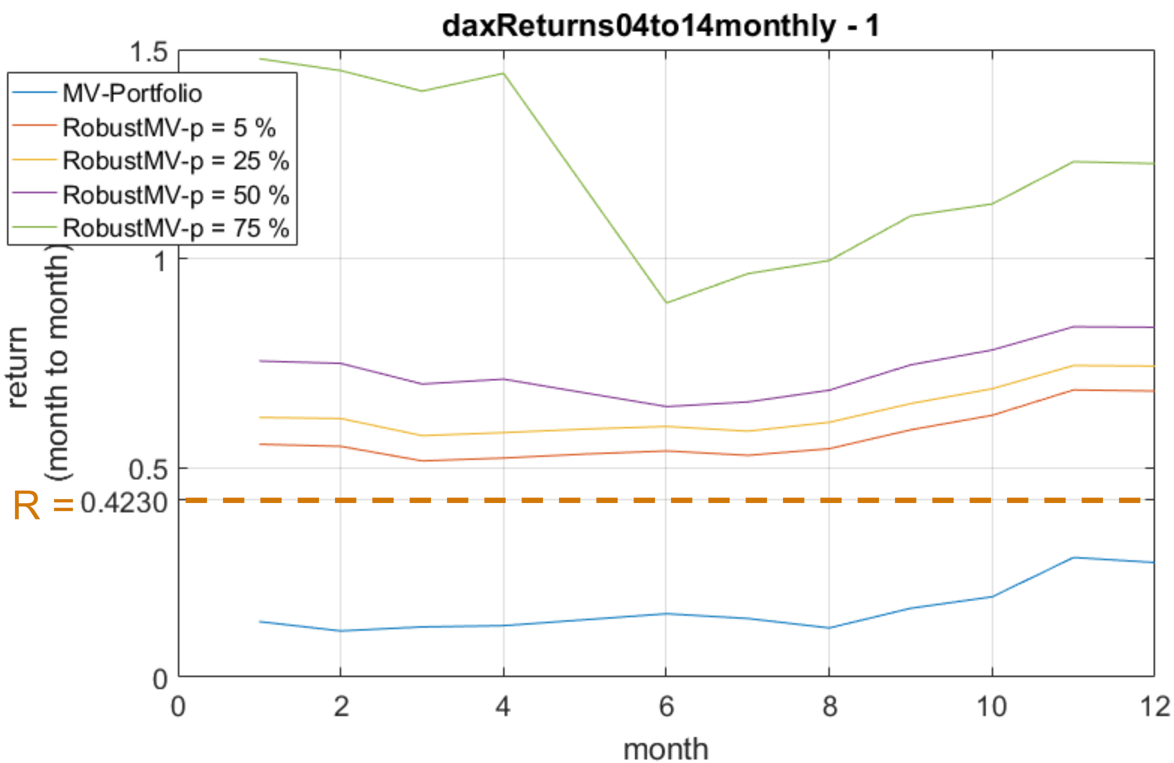
With gradients



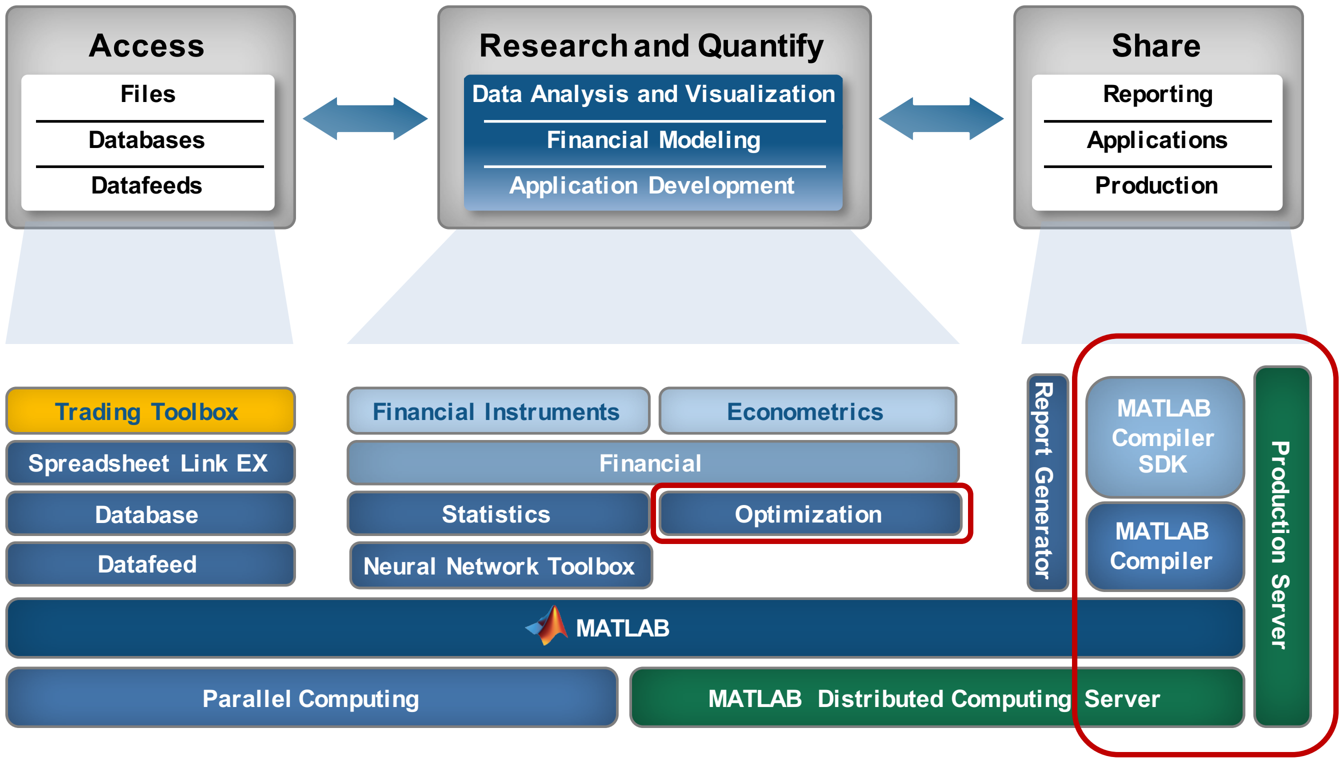
Customized Portfolio Optimization - Robust Constraints



Customized Portfolio Optimization



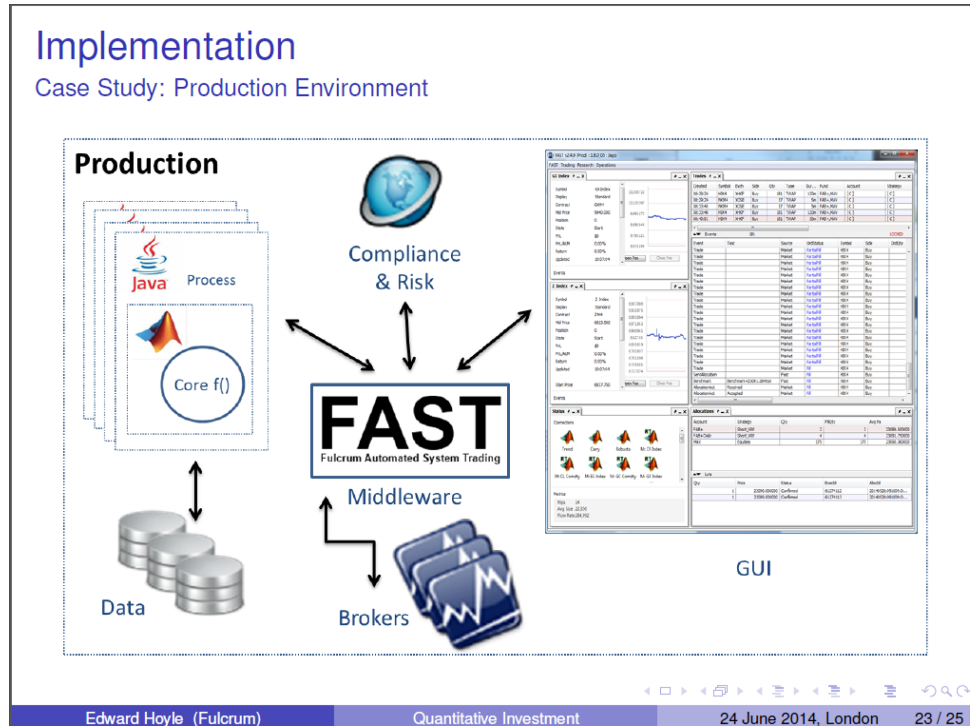
MATLAB – The Financial Development Platform



Customized Portfolio Optimization - Deployment

- Compile your MATLAB optimization model for your dedicated platform

- Make it available for your enterprise environment



MathWorks

Summary

- Optimization for financial applications is built within MATLAB toolboxes covering many standard applications

- A large variety of optimization algorithms available in MATLAB® Optimization Toolbox™ and Global Optimization Toolbox™

- Customized optimization models made easy by quick modeling, advanced optimization process diagnostics and rapid deployment.

- Enhance optimization performance and accuracy by adding maximal information.

Thank you !