Optimization in Finance: Linear and Nonlinear Programming

Research Seminar (2011)

Stefan Kerbl

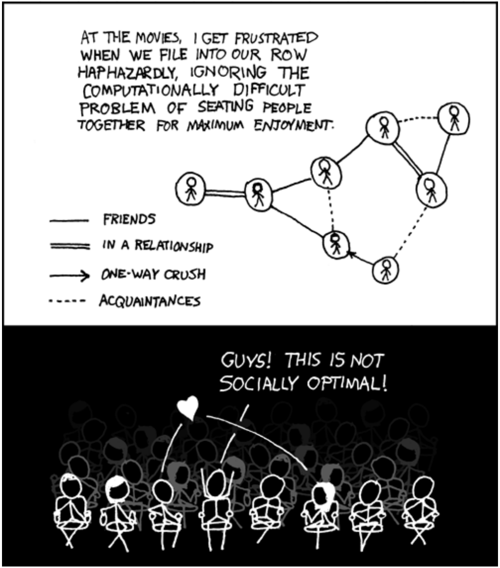
13th May 2011

Motivation

Applications of optimizing in finance

- asset liability management
- portfolio selection (mean variance optimization)
- risk management and again portfolio selection (actually the same: only variance substituted by a different risk measure)
- pricing of options and hedging with derivatives, finding of arbitrage possibilities
- via the intermediate purpose of statistics (e.g. ML-estimation)

Application in other fields



Linear Programming

The standard form LP

$$\begin{array}{lrcl} \min_x & c^T x & & \\ & Ax & = & b \\ & x & \geq & 0. \end{array}$$

tricks to get any LP in standard form:

- $\min_x c^T x = \max_x -c^T x$
- $\leq, \geq$ constraints can be coped by introducing "surplus variables"
- if x_1 can be negative or positive, define $x_1 := y_1 - y_2$ with $y_1, y_2 \geq 0$
- ...

The dual problem

$$
\begin{array}{lllll}
\max_{y,s} & b^T y & & & \\
 & A^T y & + & s & = & c \\
 & & & s & \geq & 0,
\end{array}
$$

c on the right hand side, b in the objective function
aim is to find a bound to the objective function
$\Rightarrow$ identify feasible solutions as optimum:
if primal and dual solutions are feasible and the objective values are equal, then the solution is optimal.

The simplex method

in short:

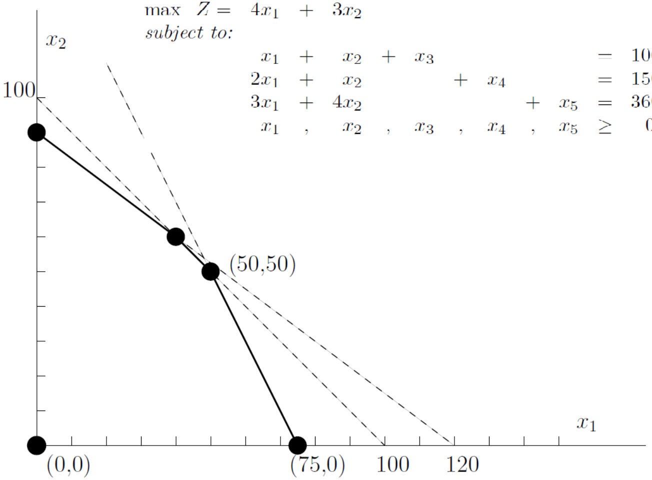
1. Split variables into *dependent* and *independent*
2. set independent to zero (basic feasible solution) and get induced values for dependent.
3. move variable with highest improvement in objective value from independent to dependent (leave basis).
4. Once all marginal effects are negative you have the optimum. (Simplex Iterations).

features:

- finite number of basic feasible solutions $\rightarrow$ eventually simplex iterations will get to optimum.
- classical approach "'The Simplex Tableau"'
- algorithm: p. 33-37 in [Cornuejols and Ttnc, 2007].

Example with graphical interpretation

$$
\begin{array}{llllllll}
\max \quad Z = & 4x_1 & + & 3x_2 & & & & \\
\textit{subject to:} & & & & & & & \\
& x_1 & + & x_2 & + x_3 & & & = 100 \\
& 2x_1 & + & x_2 & & + x_4 & & = 150 \\
& 3x_1 & + & 4x_2 & & & + x_5 & = 360 \\
& x_1 & , & x_2 & , x_3 & , x_4 & , x_5 & \geq 0.
\end{array}
$$



Simplex in R

`package:` `Rglpk`
`command:` `Rglpk_solve_LP`
`features: min/max, equality/inequalities, integer/continuous/binary, bounded/unbounded`

+ easy to use
+ flexible

- crashes if `obj` has not the respective dim as `mat`
- no direct feedback if problem unbound:
  status code not interpretable (see examples)
- no sensitivity output like `SOLVER`
- no R code → black box, hard to personalize, adapt, replicate

Nonlinear Programming

Introduction

Where does nonlinearity come from?:

1. multiplication of variables: e.g. through economics of scale, transaction costs
2. quadratic terms: variances, interest rates mean-variance optimization
3. nonlinear functions: e.g. quantiles in common risk measures

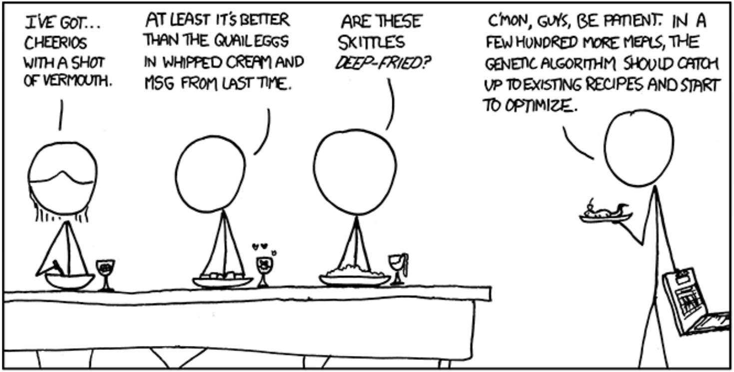
Standard representation:

$$\begin{array}{llll} \min_x & f(x) & & \\ & g_i(x) & = & 0, \quad i \in \mathcal{E} \\ & g_i(x) & \geq & 0, \quad i \in \mathcal{I}. \end{array}$$

Two simple univariate solutions

1. **binary search**:
differentiates function and then searches roots for $f'(x)$:
find 2 points with $f(A) < 0$ and $f(B) > 0$
calculate $C = (A + B)/2$ and replace A or B dependent on $f(C)$
repeat.
2. **golden section**:
divide feasible space into 4 intervals, and successively drop the interval that has the lowest outer value
choice of "golden ratio" for intermediate intervals optimizes algorithm's speed

Number of iterations possibly high..



WE'VE DECIDED TO DROP THE CS DEPARTMENT FROM OUR WEEKLY DINNER PARTY HOSTING ROTATION.

Newton's Method: finding roots

multivariate method
modified method of "steepest decent", that takes second derivative into account:
first order Taylor series approximation to find root of (system of) function(s):

$$F(\mathbf{x}^k + \delta) \approx \hat{F}(\delta) := F(\mathbf{x}^k) + \nabla F(\mathbf{x}^k)\delta$$

how to choose δ to find root?

$$\delta = -\nabla F(\mathbf{x}^k)^{-1} F(\mathbf{x}^k)$$

Newton's iteration:

$$\mathbf{x}^k + \delta = \mathbf{x}^k - \nabla F(\mathbf{x}^k)^{-1} F(\mathbf{x}^k)$$

Newton's Method: optimizing

for optimizing a function $g(x)$, $F(x)$ equals $\nabla g(x)$
the Jacobian Matrix $\nabla F(x)$ becomes the Hessian $\nabla^2 g(x)$
Hessian matrix is expensive to compute $\rightarrow$ various approximations, "quasi-Newton methods"
among them: BFGS and DFP

Newton's Method in R

optim

- `method = c(''Nelder-Mead'', ''BFGS'', ''CG'', ''L-BFGS-B'', ''SANN'')`
- `''L-BFGS-B''` allows to set bounds on parameters
- but no general optimization under constraints!
- gradient and hessian matrix returned,
- manages to overcome local minima (see example)
- very flexible, yet does not always find minima

nlm

- use of Newton's Method,
- else: unclear distinction to optim
- gradient and hessian matrix returned,
- manages to overcome local minima (see example)
- but: gets stuck at first minima!

Further Methods in R

uniroot

- Fortran Code (Newton's method?)
- if no (or any even number of) root(s) is found, error is returned $\rightarrow$ bad to handle
- else turned out to be able to handle quite complex functions

R Applications in Finance

Portfolio Optimization

fPortfolio Rmetrics

tawny

Risk Management

QRMlib

..

Further

55 findings of “optim” in package search

20 findings of “maxim” in package search

Cornuejols, G. and Ttnc, R. (2007).
Optimization Methods in Finance, volume 1 of *Mathematics, Finance and Risk*.
Cambridge University Press.