R bootcamp in Beg-Meil

Basic optimization with R

State of the R

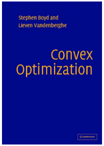
https://github.com/finistR2018

Beg-Meil, August 2018

Outline

References

See Chapter 9 in Convex Optimization (Boyd & Vandenberghe, 2004),
http://:web.stanford.edu/~boyd/cvxbook/



Online courses

All slides stolen (extracted/re-arranged) from Lieve Vandenberghe:

- Convex Optimization:
  http://www.seas.ucla.edu/~vandenbe/ee236b/ee236b.html
- Optimization Methods for Large-Scale Systems
  http://www.seas.ucla.edu/~vandenbe/ee236c/ee236c.html

Outline

Unconstrained minimization

$$\text{minimize} \quad f(x)$$

- f convex, twice continuously differentiable (hence $\mathbf{dom}\, f$ open)
- we assume optimal value $p^\star = \inf_x f(x)$ is attained (and finite)

unconstrained minimization methods

- produce sequence of points $x^{(k)} \in \mathbf{dom}\, f$, $k = 0, 1, \ldots$ with

$$f(x^{(k)}) \to p^\star$$

- can be interpreted as iterative methods for solving optimality condition

$$\nabla f(x^\star) = 0$$

Descent methods

$$x^{(k+1)} = x^{(k)} + t^{(k)}\Delta x^{(k)} \quad \text{with } f(x^{(k+1)}) < f(x^{(k)})$$

- other notations: $x^+ = x + t\Delta x$, $x := x + t\Delta x$
- Δx is the *step*, or *search direction*; t is the *step size*, or *step length*
- from convexity, $f(x^+) < f(x)$ implies $\nabla f(x)^T \Delta x < 0$
  ($i.e.$, Δx is a *descent direction*)

General descent method.

given a starting point $x \in \mathbf{dom}\, f$.
repeat

1. Determine a descent direction Δx.
2. *Line search.* Choose a step size $t > 0$.
3. *Update.* $x := x + t\Delta x$.

until stopping criterion is satisfied.

Line search types

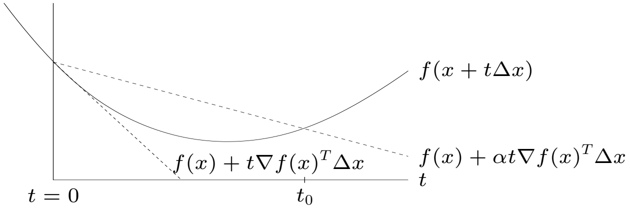
exact line search: $t = \operatorname{argmin}_{t>0} f(x + t\Delta x)$

backtracking line search (with parameters $\alpha \in (0, 1/2)$, $\beta \in (0, 1)$)

- starting at $t = 1$, repeat $t := \beta t$ until

$$f(x + t\Delta x) < f(x) + \alpha t \nabla f(x)^T \Delta x$$

- graphical interpretation: backtrack until $t \le t_0$



Outline

1. Minimization Problem
2. Gradient methods
3. Newton methods
4. Quasi-Newton methods
5. Use `stats::optim`
6. Use external library via `nloptr`
7. Use external library by embedding C++ code by yourself

Gradient descent method

general descent method with $\Delta x = -\nabla f(x)$

---

given a starting point $x \in \mathbf{dom}\, f$.
repeat
1. $\Delta x := -\nabla f(x)$.
2. *Line search.* Choose step size t via exact or backtracking line search.
3. *Update.* $x := x + t\Delta x$.

until stopping criterion is satisfied.

- stopping criterion usually of the form $\|\nabla f(x)\|_2 \leq \epsilon$
- convergence result: for strongly convex f,

$$f(x^{(k)}) - p^\star \leq c^k(f(x^{(0)}) - p^\star)$$

  $c \in (0,1)$ depends on m, $x^{(0)}$, line search type
- very simple, but often very slow; rarely used in practice

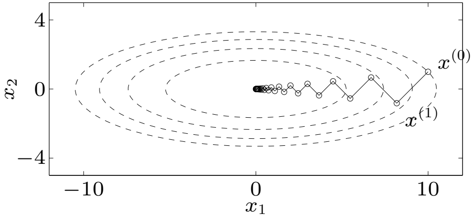
quadratic problem in $\mathbf{R}^2$

$$f(x) = (1/2)(x_1^2 + \gamma x_2^2) \qquad\qquad (\gamma > 0)$$

with exact line search, starting at $x^{(0)} = (\gamma, 1)$:

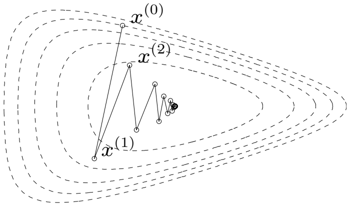
$$x_1^{(k)} = \gamma \left(\frac{\gamma - 1}{\gamma + 1}\right)^k, \qquad x_2^{(k)} = \left(-\frac{\gamma - 1}{\gamma + 1}\right)^k$$

- very slow if $\gamma \gg 1$ or $\gamma \ll 1$
- example for $\gamma = 10$:

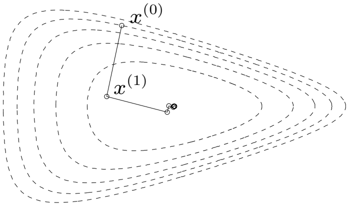


nonquadratic example

$$f(x_1, x_2) = e^{x_1+3x_2-0.1} + e^{x_1-3x_2-0.1} + e^{-x_1-0.1}$$



backtracking line search



exact line search

Outline

Newton method for unconstrained minimization

$$\text{minimize} \quad f(x)$$

f convex, twice continously differentiable

Newton method

$$x^+ = x - t\nabla^2 f(x)^{-1}\nabla f(x)$$

- advantages: fast convergence, affine invariance
- disadvantages: requires second derivatives, solution of linear equation

can be too expensive for large scale applications

Newton step

$$\Delta x_{\text{nt}} = -\nabla^2 f(x)^{-1} \nabla f(x)$$

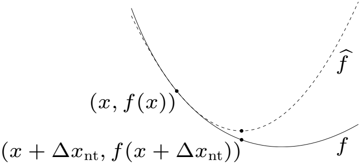
interpretations

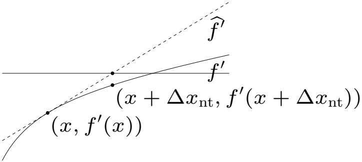
- $x + \Delta x_{\text{nt}}$ minimizes second order approximation

$$\widehat{f}(x+v) = f(x) + \nabla f(x)^T v + \frac{1}{2} v^T \nabla^2 f(x) v$$

- $x + \Delta x_{\text{nt}}$ solves linearized optimality condition

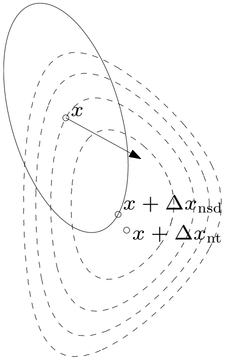
$$\nabla f(x+v) \approx \nabla \widehat{f}(x+v) = \nabla f(x) + \nabla^2 f(x) v = 0$$





- Δx_{nt} is steepest descent direction at x in local Hessian norm

$$\|u\|_{\nabla^2 f(x)} = \left(u^T \nabla^2 f(x) u\right)^{1/2}$$



dashed lines are contour lines of f; ellipse is $\{x + v \mid v^T \nabla^2 f(x) v = 1\}$

arrow shows $-\nabla f(x)$

Newton's method

---

given a starting point $x \in \mathbf{dom}\, f$, tolerance $\epsilon > 0$.
repeat

1. *Compute the Newton step and decrement.*
   $\Delta x_{\rm nt} := -\nabla^2 f(x)^{-1} \nabla f(x); \quad \lambda^2 := \nabla f(x)^T \nabla^2 f(x)^{-1} \nabla f(x)$.
2. *Stopping criterion.* **quit** if $\lambda^2/2 \leq \epsilon$.
3. *Line search.* Choose step size t by backtracking line search.
4. *Update.* $x := x + t\Delta x_{\rm nt}$.

---

affine invariant, $i.e.$, independent of linear changes of coordinates:

Newton iterates for $\tilde{f}(y) = f(Ty)$ with starting point $y^{(0)} = T^{-1}x^{(0)}$ are

$$y^{(k)} = T^{-1}x^{(k)}$$

Outline

Variable metric methods

$$x^+ = x - tH^{-1}\nabla f(x)$$

$H \succ 0$ is approximation of the Hessian at x, chosen to:

- avoid calculation of second derivatives
- simplify computation of search direction

'Variable metric' interpretation (EE236B, lecture 10, page 11)

$$\Delta x = -H^{-1}\nabla f(x)$$

is steepest descent direction at x for quadratic norm

$$\|z\|_H = \left(z^T H z\right)^{1/2}$$

Quasi-Newton methods

given starting point $x^{(0)} \in \mathrm{dom}\, f$, $H_0 \succ 0$

1. compute quasi-Newton direction $\Delta x = -H_{k-1}^{-1} \nabla f(x^{(k-1)})$
2. determine step size t (*e.g.*, by backtracking line search)
3. compute $x^{(k)} = x^{(k-1)} + t\Delta x$
4. compute H_k

- different methods use different rules for updating H in step 4
- can also propagate H_k^{-1} to simplify calculation of Δx

Broyden-Fletcher-Goldfarb-Shanno (BFGS) update

BFGS update

$$H_k = H_{k-1} + \frac{yy^T}{y^Ts} - \frac{H_{k-1}ss^TH_{k-1}}{s^TH_{k-1}s}$$

where

$$s = x^{(k)} - x^{(k-1)}, \qquad y = \nabla f(x^{(k)}) - \nabla f(x^{(k-1)})$$

Inverse update

$$H_k^{-1} = \left(I - \frac{sy^T}{y^Ts}\right) H_{k-1}^{-1} \left(I - \frac{ys^T}{y^Ts}\right) + \frac{ss^T}{y^Ts}$$

- note that $y^Ts > 0$ for strictly convex f; see page 1-9
- cost of update or inverse update is $O(n^2)$ operations

Limited memory quasi-Newton methods

main disadvantage of quasi-Newton method is need to store H_k or H_k^{-1}

Limited-memory BFGS (L-BFGS): do not store H_k^{-1} explicitly

- instead we store the m (*e.g.*, $m = 30$) most recent values of

$$s_j = x^{(j)} - x^{(j-1)}, \qquad y_j = \nabla f(x^{(j)}) - \nabla f(x^{(j-1)})$$

- we evaluate $\Delta x = H_k^{-1} \nabla f(x^{(k)})$ recursively, using

$$H_j^{-1} = \left(I - \frac{s_j y_j^T}{y_j^T s_j} \right) H_{j-1}^{-1} \left(I - \frac{y_j s_j^T}{y_j^T s_j} \right) + \frac{s_j s_j^T}{y_j^T s_j}$$

  for $j = k, k-1, \ldots, k-m+1$, assuming, for example, $H_{k-m}^{-1} = I$

- cost per iteration is $O(nm)$; storage is $O(nm)$

Outline

optim usage

Definition

```
optim(par, fn, gr = NULL, ...,
      method = c("Nelder-Mead", "BFGS", "CG", "L-BFGS-B", "SANN",
                 "Brent"),
      lower = -Inf, upper = Inf,
      control = list(), hessian = FALSE)
```

Example: Multivariate Gaussian loglikelihood

Minimize the negative MVN loglikelihood

Let $\mathbf{X}$ be a $n \times p$ data matrix (n observations, p variables), $\mathbf{S}$ the mepirical covariance matrix, $\mathbf{X}_i \sim \mathcal{N}(\mathbf{0}, \mathbf{\Sigma})$, then maximizing the likelihood w.r.t. $\mathbf{\Sigma}$ is equivalent to minimize

$$\ln |\mathbf{\Sigma}| + \mathrm{trace}(\mathbf{S}\mathbf{\Sigma}^{-1})$$

with gradient function (differentiating w.r.t. $\mathbf{\Sigma}^{-1}$)

$$-\mathbf{\Sigma}^{-1}| + \mathbf{S}$$

and Hessian $p^2 \times p^2$ matrix

$$\mathbf{\Sigma}^{-1} \otimes \mathbf{\Sigma}^{-1}$$

Example: Multivariate Gaussian loglikelihood

```
fr <- function() {}
```

Outline

Outline

References

Boyd, S., & Vandenberghe, L. (2004). *Convex optimization*. Cambridge university press. Retrieved from http://web.stanford.edu/~boyd/cvxbook/