Optimization: Uphill simplex method

Nelder & Mead (1965) Computer Journal 7:308-313
Numerical Recipes in C, §10.4

Why mention this algorithm?

• It’s cute.
• Nelder (as in McCullagh and Nelder) is a very interesting statistician.
• The method is completely different from the others we’ve discussed (as are its convergence properties).

Other points:

• Only requires function evaluations
• Not very fast
• Requires a lot of function evaluations
• Best when you need to get something going quickly and when function evaluations are cheap

Basic description:

• Seek to maximize a function $l(\theta)$ where $\theta$ is a $p$-vector.
• Start with $p + 1$ points defining a simplex in $p$-space.
• Roll/stretch/contract the simplex through $p$-space to find the maximum of $l(\theta)$
Uphill simplex method (continued)

Start:
\[ \theta_0 = \text{starting point} \]
\[ \theta_i = \theta_o + \lambda_i e_i \]
where \( e_i \) is the unit vector in the \( i \)th direction
and \( \lambda_i \) is the scale of \( \theta_{0i} \).

Move A: (reflect and expand)
Suppose \( i^* = \arg\min_{i} l(\theta_i) \).
Replace \( \theta_{i^*} \) by reflecting it across the opposite face and
expanding by a factor of 2, to give \( \theta'_{i^*} \).
Uphill simplex method (continued)

**Move B:** (reflect and contract)
If \( l(\theta_i') < l(\theta_i) \):
   - Try reflecting \( \theta_i \) across the opposite face but
     contracting by a factor of 2, to give \( \theta_i'' \).

   \[ \theta_2 \]

\[ \theta_0 \rightarrow \theta_1 \]
\[ \theta_2'' \]

**Move C:** (multiple contraction)
If also \( l(\theta_i'') < l(\theta_i') \):
   - Find \( i^{**} = \arg\max_i l(\theta_i) \)
   - Contract all points except \( \theta_i^{**} \) towards \( \theta_i^{**} \) by a factor
     of 2.

   \[ \theta_2'' \]

   \[ \theta_0 \rightarrow \theta_1 \]

**Stopping criterion:**
Stop when \( |l(\theta_i^{**}) - l(\theta_i)| < \left\{ |l(\theta_i^{**})| + |l(\theta_i)| \right\}/(2\epsilon) \)
   or maybe \( \max\{||\theta_i - \theta_i^{**}||\} < \epsilon \)
$L_p$ regression

We wish find $\hat{\beta}$ minimizing $S_p(\beta) = \sum_i |y_i - x_i'\beta|^p$

for $0 < p < 2$.

Special case: when $p = 1$ we have least absolute deviations regression

**IRLS method:**

Note that $S_p(\beta) = \sum_i w_i(\beta)(y_i - x_i'\beta)^2$

where $w_i(\beta) = |y_i - x_i'\beta|^{p-2}$

This suggests using IRLS:

1. Find a starting point $\hat{\beta}^{(0)}$ (eg, by least squares)
2. Form the weights $w_i(\hat{\beta}^{(s)}) = |y_i - x_i'\hat{\beta}^{(s)}|^{p-2}$
3. Get new estimates $\hat{\beta}^{(s+1)}$ by using least squares with weights $w_i(\hat{\beta}^{(s)})$.

**Problem:** 0 residuals

**Solution:** Take $w_i(\beta) = \max\{\epsilon, |y_i - x_i'\beta|\}^{p-2}$

for some small $\epsilon$ (eg, $10^{-8}$).

This isn’t a very stable or fast solution (though it does work, pretty much). A better solution for the case $p = 1$ will be shown later.
\textbf{$L_p$ regression: code}

\begin{verbatim}
lp <- function(x,y,p=1,tol=1e-6,eps=1e-12,maxit=1000) {
  beta.old <- lm(y~x)$coef

  for(i in 1:maxit) {
    r <- abs(y-cbind(1,x)%*%beta.old)
    r[r < eps] <- eps
    w <- as.numeric(r^(p-2))

    beta <- lm(y~x,weights=w)$coef

    if(all(abs(beta-beta.old) < 
        tol*(abs(beta.old)+tol*100))) break;

    beta.old <- beta
  }

  cat("Number of iterations:", i, \\
       "\n")
  beta
}

> print(a <- lm(y~x)$coef,dig=2)
9.26  0.30 -0.46  0.24  0.45

> unix.time(b <- lp(x,y))
Number of iterations: 41
[1] 4.33 0.52 4.91 0.00 0.00

> print(b,dig=2)
9.45  0.11 -0.44  0.29  0.51
\end{verbatim}
Constrained optimization

Maximize $l(\theta)$ for $\theta \in \Theta \subset \mathbb{R}^p$

**Easiest approach:**
Hope/pray that $\hat{\theta} = \arg \max_{\theta \in \mathbb{R}^p} l(\theta)$ satisfies $\hat{\theta} \in \Theta$.

**Usual situation:**
Maximize $l(\theta)$ subject to
- Equality constraints: $c_1(\theta) = 0, \ldots, c_E(\theta) = 0$
- Inequality constraints: $c_{E+1}(\theta) \geq 0, \ldots, c_{E+I}(\theta) \geq 0$

**Linear programming:**
- $l$ and the $c_j$ are linear in $\theta$.

**Quadratic programming:**
- $l$ is quadratic; the $c_j$ are linear.

**Nasty enough:**
- $l$ is more complex than quadratic; the $c_j$ are linear.
LAD regression

A linear programming formulation for the case $p = 1$:

Minimize $\sum e_i^+ + \sum e_i^-$

Subject to the constraints

$$y = X\beta + e^+ - e^-$$

$$e^+ \geq 0$$

$$e^- \geq 0$$

Here we have $2n + p$ unknowns (namely $\beta$, $e^+$ and $e^-$).
Since $y_i = x_i\beta + (e_i^+ - e_i^-)$, at the solution either $e_i^+ = 0$ or $e_i^- = 0$ or both.

Efficient implementations make use of the relationships among the $2n + p$ variables.

LS w/ linear equality constraints

[Seber (1977) Linear regression analysis. Wiley. §3.9.]

Minimize \((y - X\beta)'(y - X\beta)\) with constraint \(A\beta = c\)
where \(X\) is \(n \times p\) with rank \(p\)
and \(A\) is \(q \times p\) with rank \(q < p\).

Solution using Lagrange multipliers:

Consider \(r = (y - X\beta)'(y - X\beta) + (\beta' A' - c')\lambda\)

Note that \(\partial(\beta'a)/\partial\beta = a\) and
\(\partial(\beta'W\beta)/\partial\beta = 2W\beta\) (if \(W\) is symmetric).

Thus \(\partial r/\partial\beta = 2X'y + 2X'X\beta + A'\lambda.\)

We seek \(\hat{\beta}_H\) and \(\hat{\lambda}_H\) satisfying \(A\hat{\beta}_H = c\)
and \(2X'y + 2X'X\hat{\beta}_H + A'\hat{\lambda}_H = 0.\)

\[
\hat{\beta}_H = (X'X)^{-1}X'y - (X'X)^{-1}A'\hat{\lambda}_H/2 \\
= \hat{\beta} - (X'X)^{-1}A'\hat{\lambda}_H/2 \\
c = A\hat{\beta} - A(X'X)^{-1}A'\hat{\lambda}_H/2
\]

\(\Rightarrow\) \(\hat{\lambda}_H/2 = \{A(X'X)^{-1}A'\}^{-1}(A\hat{\beta} - c)\)

\(\Rightarrow\) \(\hat{\beta}_H = \hat{\beta} + (X'X)^{-1}A'\{A(X'X)^{-1}A'\}^{-1}(c - A\hat{\beta})\)
LS w/ lin eq constraints (continued)

Three other points:
1. An interesting geometric derivation of $\hat{\beta}_H$, using projections and nullspaces, appears in Seber (1977) §3.9, and is worth looking at.

2. Since $\hat{\beta}_H = G\hat{\beta} + k$, we have $\text{var}(\hat{\beta}_H) = G \text{var}(\hat{\beta}) G'$
   Using $G = I - (X'X)^{-1}A'\{A(X'X)^{-1}A\}^{-1}A$,
   we appear to obtain $\text{var}(\hat{\beta}_H) = \sigma^2 G(X'X)^{-1}$

3. I can't remember my third point.

Example
Consider the model $y = \beta_0 + \sum_{i=1}^{4} \beta_i x_i + \epsilon$
Suppose we have the constraints $2\beta_1 = \beta_4$ and $3\beta_2 = -5\beta_3$.
Then $c = 0$ and 

$$A = \begin{pmatrix} 0 & 2 & 0 & 0 & -1 \\ 0 & 0 & 3 & 5 & 0 \end{pmatrix}$$

With the data I played with above in LAD regression,
\( \hat{\beta} \approx (9.26, 0.30, -0.46, 0.24, 0.45) \) and \( \hat{\beta}_H = G\hat{\beta} \approx (9.30, 0.23, -0.45, 0.25, 0.47) \).

\( \hat{SE}(\beta) \approx (0.59, 0.14, 0.12, 0.14, 0.11) \)  
\( \hat{SE}(\beta_H) \approx (0.51, 0.03, 0.12, 0.07, 0.07) \)
LS with linear inequality constraints

This is an example of quadratic programming.

Note:
A consistent set of linear inequality constraints of full rank can be reduced by reparameterization to a set of nonnegativity constraints. [Thisted, 1988]

Problem:
minimize $(y - X\beta)'(y - X\beta)$
with constraints $\beta_j \geq 0$ for $j \in J$

Algorithm:
Let $\hat{\beta}$ be the unconstr’d sol’n, $C = J \setminus \{j : \hat{\beta}_j \geq 0\}$ (the “active” constraints), and $M = \{1, \ldots, p\} \setminus J$.

The constr’d sol’n to the problem is equiv’t to the unconstr’d sol’n dropping the columns of $X$ with $j \in C$.

Thus to solve the constrained problem, we need to find the maximal set $M$ for which the unconstrained estimates $\hat{\beta}_j$ all satisfy the constraints.

At the worst, we may need to fit $2^{|J|}$ models. But a step-wise selection procedure can get the job done.
Example

Consider the model \( y = \beta_0 + \sum_{j=1}^{4} x_j + \epsilon \) where we constrain \( \beta_j \geq 0 \) for \( j > 0 \).

\[
\begin{align*}
> \text{lm}(y \sim x) \text{\$coef} \\
& \text{int \hspace{0.5cm} X1 \hspace{0.5cm} X2 \hspace{0.5cm} X3 \hspace{0.5cm} X4} \\
& -0.0822 \hspace{0.5cm} -0.0388 \hspace{0.5cm} 0.1221 \hspace{0.5cm} 0.0016 \hspace{0.5cm} 0.1226
\end{align*}
\]

\[
\begin{align*}
> \text{lm}(y \sim x[,,-1]) \text{\$coef} \\
& \text{int \hspace{0.5cm} X2 \hspace{0.5cm} X3 \hspace{0.5cm} X4} \\
& -0.0867 \hspace{0.5cm} 0.1071 \hspace{0.5cm} -0.0107 \hspace{0.5cm} 0.1271
\end{align*}
\]

\[
\begin{align*}
> \text{lm}(y \sim x[,,-3]) \text{\$coef} \\
& \text{int \hspace{0.5cm} X1 \hspace{0.5cm} X2 \hspace{0.5cm} X4} \\
& -0.0823 \hspace{0.5cm} -0.0384 \hspace{0.5cm} 0.1228 \hspace{0.5cm} 0.1225
\end{align*}
\]

\[
\begin{align*}
> \text{lm}(y \sim x[,,-c(1,3)]) \text{\$coef} \\
& \text{int \hspace{0.5cm} X2 \hspace{0.5cm} X4} \\
& -0.0861 \hspace{0.5cm} 0.1009 \hspace{0.5cm} 0.1289
\end{align*}
\]

The last model, with \( \hat{\beta}_1 = \hat{\beta}_3 = 0 \), is the one we choose.
Another example: Isotonic regression

Consider pairs \((x_i, y_i)\) with \(x_1 \leq x_2 \leq \ldots \leq x_n\).

Suppose \(y_i|x_i \sim \mathcal{N}(\mu_i, \sigma^2)\) with \(\mu_1 \leq \mu_2 \leq \ldots \leq \mu_n\).

→ Find \(\hat{\mu}_i\) minimizing \(\sum (y_i - \mu_i)^2\) with this constraint.

If \(y_1 \leq y_2 \leq \ldots \leq y_n\), then things are easy: \(\hat{\mu}_i = y_i\!\).

Otherwise:

1. Let \(\beta_1 = \mu_1\) and \(\beta_i = \mu_i - \mu_{i-1}\) for \(i > 1\).
   
   Let \(X_{ij} = 1\) if \(i \geq j\) or 0 otherwise.

   Then \(y = X\beta + \epsilon, \ \epsilon \sim \mathcal{N}(0, \sigma^2), \ \beta_i \geq 0\) for \(i > 1\).

   Use the previously described method!

2. “Pool adjacent violators”
   
   See Barlow, Bartholomew, Bremner and Brunk (1972) Statistical inference under order restrictions; the theory and application of isotonic regression. Wiley.
Example

Suppose $k_{ij} \sim \text{indep Poisson}(\lambda_i)$ for $i = 1, \ldots, G$ and $j = 1, \ldots, n_i$.

$$l(\lambda|k) = -\sum_i n_i \lambda_i + \sum_i \log \lambda_i \sum_j k_{ij}$$

$$\partial l/\partial \lambda_i = -n_i + \sum_j k_{ij}/\lambda_i$$

**MLE:** $\hat{\lambda}_i = \sum_j k_{ij}/n_i$

**Constraint:** $\lambda_1 \leq \lambda_i$ for all $i$.

If $\hat{\lambda}_1 \leq \hat{\lambda}_i$ for all $i \rightarrow$ done!

Otherwise, suppose $\hat{\lambda}_i < \hat{\lambda}_1$ for $i \in I$.

Then $\hat{\lambda}'_1 = \sum_{i \in I \cup \{1\}} \sum_j k_{ij}/\sum_{i \in I \cup \{1\}} n_i$ and $\hat{\lambda}'_i = \hat{\lambda}'_1$ for $i \in I$
Nonquad programming w/ lin eq constr

We seek to maximize $l(\theta)$ with the constraint $A\theta = b$ where $A$ is $q \times p$ with rank $q < p$.

Let $Z$ be a $p \times (p - q)$ orthonormal matrix satisfying $AZ = 0$ (and $Z'Z = I$).

Basic algorithm:

1. **Start**: Pick $\hat{\theta}^{(0)}$ satisfying $A\hat{\theta}^{(0)} = b$.

2. **Steps**: Take $\hat{\theta}^{(s+1)} = \hat{\theta}^{(s)} + \alpha_s \delta_s$ where $\delta_s = Zy$ for some $(p - q)$-vector $y$.

Let $g^{(s)}$ and $G^{(s)}$ be the gradient and Hessian of $l$, respectively, evaluated at $\hat{\theta}^{(s)}$.

**Steepest ascent**: $\delta_s = ZZ'g^{(s)}$

**Newton-Raphson**: $\delta_s = -ZZ'G^{(s)}Z^{-1}Z'\hat{g}^{(s)}$

Nonquad programming w/ lin ineq constr

We seek to maximize \( l(\theta) \) with the constraint \( A\theta \geq b \)
where \( A \) is \( q \times p \) of rank \( q < p \).

The following is what is called an active set algorithm.

1. Find a starting point \( \hat{\theta}^{(0)} \) satisfying \( A\theta \geq b \). (For example, find the unconstrained maximum and project into onto the feasible set \( \{\theta : A\theta \geq b\} \).

2. Define the set of active constraints by \( C_s = \{i : a_i'\theta^{(s)} = b_i\} \) where \( a_i' \) is the \( i \)th row of \( A \).
Let \( A_s \) and \( b_s \) denote the \( A \) and \( b \) with the inactive constraints dropped.

3. Consider dropping a constraint from the active set.

4. Compute a feasible direction, \( \delta_s = Z_s y \) where \( A_s Z_s = 0 \) and \( Z_s' Z_s = I \), as if we were maximizing \( l(\theta) \) with just the active constraints \( A_s \theta = b_s \).

5. Calculate \( \bar{\alpha}_s \), the maximum value of \( \alpha \) such that \( A(\hat{\theta}^{(s)} + \alpha \delta_s) \geq b \)
Find \( 0 < \alpha_s \leq \bar{\alpha}_s \) so that \( l(\hat{\theta}^{(s)} + \alpha_s \delta_s) > l(\hat{\theta}^{(s)}) \).

6. Take \( \hat{\theta}^{(s+1)} = \hat{\theta}^{(s)} + \alpha_s \delta_s \).

7. If \( \alpha_s = \bar{\alpha}_s \), add the corresponding inactive constraint to the active set.