771: A Crash Course in Convex Optimization

Notes from Oct 2 and Oct 4, 2023

We survey some important facts from convex analysis as they relate to statistical computations. We'll be mathematical, but our proofs are a bit like winter coats. We'll throw them on to *keep warm*, but won't fully zip them up! The hope is that in one or two lectures we can get some useful mathematical tools in view. The theory of convex analysis is broad and deep; students are encouraged to use these brief notes as a light entrez to a more fullfilling investigation. See the syllabus for citations to useful books on convex analysis.

Convex sets.

A subset C of Euclidean space (R^d) is convex if for all $x, y \in C$, all the points on the line segment joining x and y are also in C.

```
[examples]
```

(not mentioned in class, but); translations and rotations (and any affine maps) of a convex set are convex. Intersections are convex, but not necessarily unions.

Hyperplanes.

A non-zero point $a \in R^d$ and another $b \in R$ define a hyperplane $H = \{x : \ a^T x = b\}$. This is an $d-1$ dimensional subset of R^d. Taking any point $x_0 \in H$, we note that H is the set of x's formed by adding x_0 to v, where $a^T v = 0$, and so the hyperplane is a shifted version of vectors orthogonal to a.

Halfspaces:

sets such as $\{x : a^T x \leq b\}$ (they may be open or closed, depending on if inequality is strict or not)

Separating hyperplane theorem

The first cool convex analysis result:

For any two non-empty disjoint convex sets C and D, there is a plane $H_{a,b}$ such $a^T x \leq b$ for all $x \in C$ and $a^T x \geq b$ for all $x \in D$.

From this one may derive the

Supporting hyperplane theorem

For any convex set C and any point x_0 on its boundary [closure minus interior], there exists a hyperplane $H = H_{a,b}$ such that $x_0 \in H$ and $a^T x \leq b = a^T x_0$ for all $x \in C$.

[You can imagine this might be true using an open C and a singleton $D = \{x_0\}$ and applying the sep hyp thm].

Convex functions.

I like the epigraph definition. Given any function f and its domain dom(f) (which we will assume is a convex set in R^d), we have a subset of $R^{d+1}

$$epi(f) = \{(x, y) : x \in \ dom(f), \ y \geq f(x)\}$$

Define f to be a convex function if $epi(f)$ is a convex set.

It turns out this is equivalent to the usual definition involving f evaluated at a point between two distinct points being less than or equal to the line segment at that intermediate point; this more standard approach also gives us the definition of *strict* convexity, as we can impose strict inequality in all cases.

Preserving convexity

Many operations preserve convexity. E.g. the sum of convex functions is convex and the pointwise max of convex functions is convex. The difference of convex functions need not be convex, though there's some interesting optimization theory available for the larger class of functions that are differences of convex functions.

Jensen's inequality.

Taking X a random variable on dom(f) with $E(X) = \mu$, then $f(X) \geq f(\mu) + c(X - \mu)$ [by the supporting hyperplane theorem on epi(f), at $(\mu, f(\mu))$]. And then take expectations to both sides: $E[f(X)] \geq f[E(X)]$. Note, the inequality is strict for strictly convex functions, as long as X is not a degenerate random variable.

Subgradients.

A vector g is a subgradient of f at x_0 if $f(x) \geq f(x_0) + g^T(x - x_0)$ for all x in dom(f).

Such things exist for convex functions thanks to supporting hyperplane theorem on epi(f), but they may not be unique [e.g. $f(x) = |x|$ in R.]

Where differentiable functions have gradients, convex functions always have subgradients. The two coincide if the convex function is differentiable. Indeed, if there's a unique subgradient g at x then f is differentiable at f and and $g = \nabla f(x)$.

The theory about subgradients provides some useful tools for optimization where calculus is not able to provide answers.

Subdifferential.

Consider the set-valued function

$$\partial f(x) = \{g : \ g \, \text{is a subgradient of f at x}\}$$

example: $\partial|x| = [-1, 1]$ at $x = 0$; equals singleton -1 if $x < 0$ and equals $+1$ if $x > 0$.

[it turns out that $\partial f(x)$ is closed and convex for convex f]

draw a few examples

Subgradient optimality.

We say x^* is a global minimimzer of f if $f(x^*) \leq f(x)$ for all x in dom(f).

Claim: x^* is a global minimizer of f if and only if $0 \in \partial f(x^*)$.

To see why, note simply that for any subgradient $g \in \partial f(x^*)$, $f(x) \geq f(x^*) + g^T(x - x^*)$ for all x, and so 0 cancels the right term and gives the optimality.

This is imporant because it gives a condition [not an algorithm] for a solution x^* to be a global optimizer.

Example: sum of absolute deviations

Consider an ordered set of distinct numbers $d_1, d_2, \cdots, d_n$. Let $f(\mu) = \sum_{i=1}^n |d_i - \mu|$. The question is to find $\hat{\mu}$ that minimizes $f(\mu)$ over the reals. Applying the ideas above, whatever is $\hat{\mu}$, it must be that $0 \in \partial f(\hat{\mu})$. Because f is a sum, any point $v \in \partial f(\mu)$ (for any μ) must be a sum of contributions v_i with $v_i \in \partial|x_i - \mu|$. And we recall, that such v_i is $+1$ if $\mu > x_i$, is -1 if $\mu < x_i$ and is some value in $[-1, 1]$ if $\mu = x_i$.

Take the even case, with $n = 2m$, and take any value $\hat{\mu} \in (d_m, d_{m+1})$. We calculate the same number of $+1$'s and -1's contributing to v, and so clearly $0 \in \partial f(\hat{\mu})$, and so $\hat{\mu}$ solves the optimization. A similar argument holds for odd n (and there's a unique median in this case!). We've shown that the sample median minimizes the sum of absolute deviations from a point.

Example: Regularized regression

Let's look again at response-predictor data: the $n \times 1$ response vector y and the $n \times p$ covariate matrix X.

We've studied the least squares problem, which finds $\hat{\beta}_{OLS}$ to minimize the objective function

$$\frac{1}{2}(y - X\beta)^T(y - X\beta)$$

and we discussed some basic numerical linear algebra to solve this. The more recently developed and much discussed LASSO method minimizes

$$f(\beta) = \frac{1}{2}(y - X\beta)^T(y - X\beta) + \lambda \sum_{j=1}^{p} |\beta_j|$$

Why we would want to solve this objective is an important question, but one we defer (basically it's responding to the need for combined model-selection model estimation schemes: which covariates ought to be in the model?).

The second term above is a so-called L_1 penalty; if λ is near 0 the penalty is weak and we expect the solution near the OLS solution. As λ increases the solution is penalized ever more for L_1 deviations from the origin. Indeed, the least squares objective is regularized in the sense that a LASSO solution will exist even if the OLS solution does not.

Note that f is convex but not differentiable [this is the so-called Lagrangian version of the LASSO problem; an alternative, *constrained* version will be discussed later].
From the discussion of subdifferentials above, we know that β^* is a solution if the zero vector is an element of the subdifferential of f at β^*:

$$0 \in \partial f(\beta^*)$$

Further, since f is the sum of two pieces, any vector g is in the subdifferential if g is a subgradient of the sum, and thus is the sum of subgradients of the two components of f. This means

$$0 = -X^T(Y - X\beta^*) + \lambda\nu$$

for some $\nu \in \partial \left[\sum_j |\beta_j|\right]$ at $\beta = \beta^*$. What can we say about the vector ν? Well it's in the subdifferential of the L_1 function, and so it's a sum of vectors, the jth of which is in the subdifferential of $|\beta_j|$. We have

$$\nu_j = \begin{cases} +1 & \text{if } \beta_j^* > 0 \\ -1 & \text{if } \beta_j^* < 0 \\ c \in [-1, 1] & \text{if } \beta_j^* = 0 \end{cases}$$

For this curious, solution-dependent vector ν, a LASSO solution β^* must therefore satisfy

$$X^T(y - X\beta^*) = \lambda\nu$$

Of course this equals the normal equations if $\lambda = 0$, but generally the solution is different from the OLS solution. Note that if $X = [x_1, x_2, \cdots, x_p]$, then the jth entry of this vector equation is

$$x_j^T(y - X\beta^*) = \lambda\nu_j$$

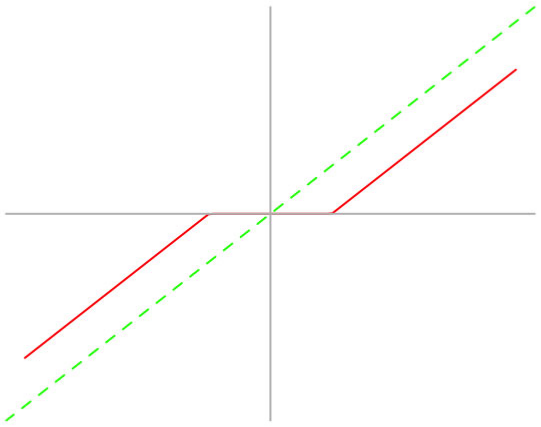
and thus by the definition of ν, it must be, for example that

$$|x_j^T(y - X\beta^*)| < \lambda$$

if and only if $\beta_j^* = 0$. This is the first glimpse of the *selection* operation of the LASSO; when a certain fit (on the left) is small, the estimated coefficient must be zero. To see this more clearly we consider two special cases.

Example. $X = I$. We reduce the LASSO equations to $\beta_i^* = y_i - \lambda$ if $y_i > \lambda$, $\beta_i^* = y_i + \lambda$ if $y_i < -\lambda$, and else $\beta_i^* = 0$. This corresponds to the *soft-threshold* function $S_\lambda(y)$ plotted below.

```
ygrid <- seq(-4,4,length=100)
plot( ygrid, ygrid, axes=FALSE, xlab="", ylab="", type="n" )
abline(0,1, lty=2, col="green" )
soft <- function(y,lambda)
     {
       tmp <- sign( y ) * (abs(y)-lambda)
       res <- ifelse( abs(y) < lambda , 0, tmp )
     }
s <- soft(ygrid,lambda=1)
lines( ygrid, s, col="red" )
abline(h=0,col="grey"); abline(v=0, col="grey" )
```



Example. $X = x_1$, and hence we have $p = 1$ predictor.
We reduce the LASSO equations here to $\beta^* = S_\lambda(\hat\beta_{OLS})$, where $\hat\beta_{OLS}$ is the ordinary least squares estimator $x_1^T y/(x_1^T x_1)$ in this case.

In these two special cases there is a closed form solution for the LASSO solution, but in general there is not. For example, one might be tempted to assume that we could soft threshold components of the ordinary vector $\hat\beta_{OLS}$, but this does not lead to the LASSO solution. Before considering just how to compute the β^* vector in general, let's take a diversion into an alternative perspective on the LASSO problem.

Indicators

In statistics, an indicator function is a $1/0$-valued function that takes value 1 for arguments in the set. There's another kind of indicator function in convex analysis.

Define $1_C(x)$ to take value 0 if $x \in C$ and value ∞ if $x \notin C$.

When minimizing a function $f(x)$ subject to the constraint $x \in C$, it is equivalent to minimize the function $f(x) + 1_C(x)$, since the infinity bit forces the argmin to land in C.

Normal cones

To see the role of indicators we first need something else:

For $x_0 \in C \subset R^n$, define the set

$$N_C(x_0) = \{g \in R^n : g^T(x - x_0) \leq 0 \quad \forall x \in C\}$$

It's a proposition [not proved] that $N_C(x_0)$ itself is a convex cone. [Recall, a cone is a set closed under multiplication by any non-negative constant.]

We can show that normal cones arise as subdifferentials of indicator functions, and this is key to a useful formula to determine optimality.

Continuing from last time, let's have a convex set $C \subset R^n$, and recall both the indicator function $1_C(x)$ and the normal cone $N_C(x_0) = \{g \in R^n : \ g^T(x - x_0) \leq 0 \ \forall x \in C\}$ introduced above.

[examples of some normal cones in simple sets, where $x_0 = 0$]

Recall at least that when $0 \in C$, $N_C(0)$ is the set of vectors g making non-positive inner product with all vectors in C.

We investigate the subdifferential of $1_C(x_0)$ for $x_0 \in C$, and note that it's the set of vectors g such that $1_C(x) \geq 1_C(x_0) + g^T(x - x_0)$ for all $x \in dom(f)$. Owing to the infinity on the left when x is not in C, we have no constraints by those values, and g is in the subdifferential if

$$0 \geq g^T(x - x_0) \qquad \forall x \in C$$

In other words, the normal cone is the subdifferential of the indicator of a convex set.

Differentiable convex functions subject to convexity constraints

We're thinking about statistical estimation problems in which the parameter estimate is minimizes a certain objective function (e.g., negative log likelihood) subject to some structural constraint on the parameters. This fits into a general class of optimization problems, some of which include nonparametric shape constraints to regularize model fits.

Say f is differentiable and convex, so an unconstrained global optimizer x^* would satisfy $\nabla f(x^*) = 0$. But we also require our solution to be restricted to a convex set C. To provide for this, it's enough that x^* be the global optimizer of the modified function

$$f(x) + 1_C(x)$$

We know from subgradient optimality that for x^* to solve this it must be that $0 \in \partial[f(x) + 1_C(x)]$. Using the addititivity property of subdifferentials and the connection just made to normal cones, any subgradient in this subdifferential must be the sum of $\nabla f(x^*)$ and some vector v in the normal cone of C at x^*. I.e., $-\nabla f(x^*)$ must be in the normal cone of C at x^*:

$$[-\nabla f(x^*)]^T(x - x^*) \leq 0 \qquad \forall x \in C$$

This important formula is sometimes called *first-order optimality*. It characterizes any solution x^* that minimizes a differentiable convex function f subject to convex constraints C.

First-order optimality may not be useful in that it's not a prescription for computing x^*. But if you have a candidate x^* you may check that is in fact a proper solution.

Curiously, this general theory covers a range of specialized cases we may know about already:

1. $C = R^n$. Then there are really no constraints, and the formula holds when x^* is chosen so that $\nabla f(x^*) = 0$, which we already knew from calculus characterizes the optimizer.
2. C is a set of linear equality constraints. Here we take an example to help us. Consider observed multinomial counts $\{y_k\}$ from n i.i.d. trials on a discrete population, with type k occuring with probability p_k. The negative log likelihood for the full vector of propertions, p, is $f(p) = -\sum_{k=1}^K y_k \log(p_k)$. We are familiar with the finding that the MLE is $\hat{p}_k = y_k/n$. This involves minimizing f subject to the constraint that $\sum_k p_k = 1$, and is readily developed using a Lagrange multiplier. On the other hand, we could consider f at first unconstrained, and then require our solution to live in the convex set $C = \{p : \sum_k p_k = 1,\ p_k \geq 0\}$. By this route, the gradient $\nabla f(\hat{p})$ has entries $-y_k/\hat{p}_k = (-n)$, and the first-order optimality condition holds immediately:

$$[-\nabla f(\hat{p})]^T(p - \hat{p}) = \sum_{k=1}^K (n)(p_k - \hat{p}_k) = 0.$$

3. C is a set of linear equality and inequality constraints. The powerful KKT theory covers this case, and provides specificity for x^*. But KKT is superceded by first-order optimality, in the sense that a KKT solution has to also be a solution to the formula above.

Example: MLE for ordered means

As an example, suppose data are $\{(i, Y_i) : i = 1, 2, \cdots, n\}$ and our model is that the measurements are independent and governed by:

$$Y_i \sim \text{Normal}(\mu_i, \sigma^2)$$

Say the observed value of Y_i is y_i.

Let $\mu = (\mu_1, \mu_2, \ldots \mu_n)^T$ be the parameter of interest. As in our treatment of regression, we'll initially ignore σ^2, since we know that the MLE of μ, regardless of σ^2, must minimize the objective:

$$f(\mu) = (1/2)\sum_{i=1}^n (y_i - \mu_i)^2$$

.

This function is proportional to the negative log likelihood. Without constraints $\hat{\mu}_i = y_i$, and we'd probably be overfitting the system. We add structure to the problem by requiring our solution to be have ordered components; i.e. to lie in the convex set C:

$$C = \{\mu :\ \mu_1 \leq \mu_2 \leq \cdots \leq \mu_n\}$$

Finding the MLE subject to $\mu \in C$ is a tricky problem, but one that the current theory provides for, since f is convex and differentiable and C is convex. Accordingly, our solution $\hat{\mu}$ must satisfy first-order optimality:

$$[-\nabla f(\hat{\mu})]^T(\mu - \hat{\mu}) \leq 0 \qquad \forall \mu \in C$$

We know further that $\nabla f(\mu)$ has components $\mu_i - y_i$, and so whatever is our solution it must satisfy:

$$\sum_{i=1}^{n} (\hat{\mu}_i - y_i)(\mu_i - \hat{\mu}_i) \geq 0 \qquad \forall \mu \in C$$

We're not out of the woods, but we've made some progress. To go further, let's first make a diversion.

Cumulative sum diagram (CSD)

Consider a two-dimensional plot of points

$$P_i = \left(i, \sum_{j=1}^{i} y_j \right) \qquad i = 1, 2, \cdots, n,$$

and including $P_0 = (0,0)$. Let $H_n(t)$ denote the function that interpolates these points. Plotting H_n creates the so-called *cumulative sum diagram*.

Greatest convex minorant (GCM)

The GCM, denoted $H(t)$, say, is the pointwise supremum of all convex functions lying below H_n, and is itself convex. Notice this function is piecewise linear.

Left-derivative of GCM

For arguments $t = i$, $i = 1, 2, \cdots, n$, compute the left derivative of $H(t)$, and denote each $\tilde{\mu}_i$. I.e.

$$\tilde{\mu}_i = \lim_{\epsilon \to 0} \left(H(i - \epsilon) - H(i) \right) / \epsilon$$

I.e., these are the slopes of the GCM taken on the left side of each position i.

Claim: The left derivatives of the GCM of the CSD (i.e., the $\tilde{\mu}_i$'s) are the maximum likelihood estimates of μ_i subject to the ordering constraint C.

Before trying to establish this claim, it's useful to have an alternative characterization of the GCM. We claim without proof, though by inspection of an example (!), that the following is true:

$$\sum_{j=1}^{i} \tilde{\mu}_j \leq \sum_{j=1}^{i} y_j \qquad \forall i$$

and further that the equality here is strict if $\tilde{\mu}_{i+1} > \tilde{\mu}_i$ or if $i = n$. (The equality *may* be strict otherwise but must be strict in the cases noted.)

[sketch the CSD and GCM in one case]

In class we work through the toy example $y = (0, 1, 1, 0, 0, 1, 1, 1)$.

Proof that left-derivative of GCM of CSD satisfies first-order optimality

The first-order optimality criteria would hold for the $\tilde{\mu}_i$'s if

$$[-\nabla f(\tilde{\mu})]^T (\mu - \tilde{\mu}) \leq 0 \qquad \forall \mu \in C$$

Recall that the gradient has entries $\tilde{\mu}_i - y_i$, and so the critical value (that we aim to show is non-negative) is

$$\sum_{i=1}^{n} (\tilde{\mu}_i - y_i)(\mu_i - \tilde{\mu}_i) = \sum_{i=1}^{n} d_i \mu_i - \sum_{i=1}^{n} d_i \tilde{\mu}_i$$

where $d_i = \tilde{\mu}_i - y_i$. Call this $A - B$, and work on each piece separately.

To sort either, notice the ordering constraint means that any $\mu \in C$ may be expressed in terms of other parameters $\alpha_0, \alpha_1, \cdots, \alpha_{n-1}$, with $\alpha_j \geq 0$ for $j = 1, 2 \cdots, n-1$. These are the jumps associated with increases in μ_i. Therefore

$$\mu_i = \alpha_0 + \sum_{j=1}^{n-1} \alpha_j 1[j < i]$$

Thus

$$A = \sum_{i=1}^{n} d_i \left(\alpha_0 + \sum_{j=1}^{n-1} \alpha_j 1[j < i] \right)$$

Simplifying, and switching the summation order

$$A = \alpha_0 \sum_{i=1}^{n} d_i + \sum_{j=1}^{n-1} \alpha_j \sum_{i=1}^{n} d_i 1[j < i]$$

The characterization of the $\tilde{\mu}_i$'s (specifically, the $i = n$ case) proves that $\sum_{i=1}^{n} d_i = 0$. This gives

$$A = \sum_{j=1}^{n-1} \alpha_j \left(- \sum_{i=1}^{j} d_i \right)$$

Both components in the sum over j are non-negative, since $\alpha_j \geq 0$ by the ordering constraint and the sum in brackets is non-negative by the characterization.

It remains to prove that $B = 0$. Expanding from above,

$$B = \sum_{i=1}^{n} d_i \left(\tilde{\alpha}_0 + \sum_{j=1}^{n-1} \tilde{\alpha}_j 1[j < i] \right)$$

where the $\tilde{\alpha}_j$'s are deviations associated with $\tilde{\mu}$. Simplifying

$$B = \sum_{j=1}^{n-1} \tilde{\alpha}_j \left(\sum_{i=j+1}^{n} d_j \right) = - \sum_{j=1}^{n} \tilde{\alpha}_j \left(\sum_{i=1}^{j} d_i \right)$$

. Now we consider the two factors in each summand on the right side above. From the characterization of the $\tilde{\mu}_i$'s , each $\tilde{\alpha}_j$ is either 0 or positive. The key is to observe that when it's positive, the other factor $\sum_{i=1}^{j} d_i$ equals zero, thus completing the proof.

Pool Adjacent Violators

PAVA is a step-wise algorithm that ends in a finite number of steps at the exact solution (in contrast to other iterative algorithms that we'll see later which continue until some tolerance is met).

One way to describe it is to say at each step `t` there are sets S1, S2, …, SJ constituing contiguous sets of integers in `1:n` , there are *tentative* estimates muhat_{sj}, and weights w_{Sj}:

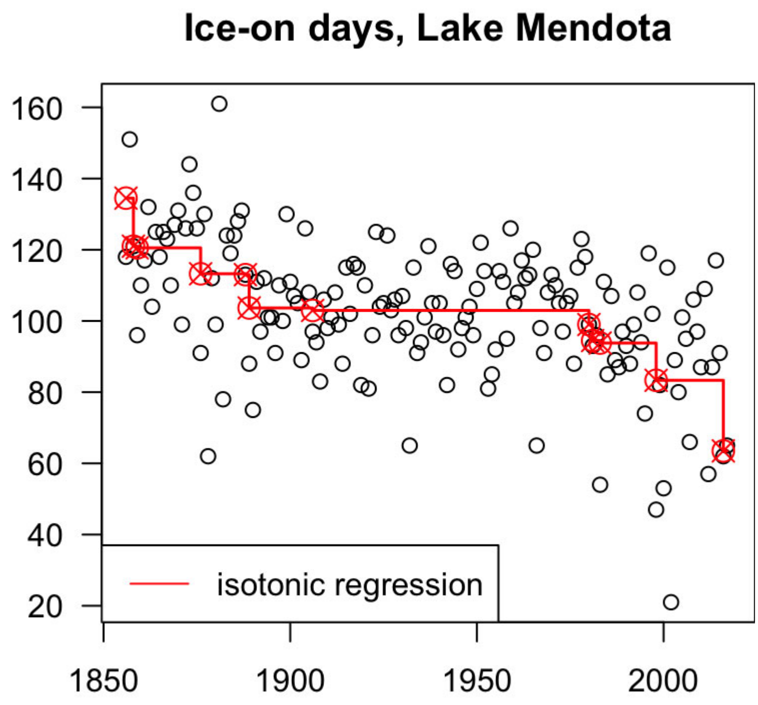
```
Initialize:  set Sj = {j}; weight w_j = 1, and estimate muhat_{Sj} = y_j.

A. Find adjacent violators. I.e.,  where
         muhat_{Sj} > muhat_{S(j+1)}

If no violators: stop.

If adjacent violators:

Pool by computing:
        bar <- [w_{Sj}+w_{S(j+1)}]
        tmp <- [w_{Sj} muhat_{Sj} + w_{S(j+1)} muhat_{S(j+1)} ]/bar

Update the system:

                merge   sets  Sk <- Sk  k < j
                              Sj <- Sj union S(j+1)
                              Sk <- S(k+1) , k >= j+1

                compute means muhat_{Sk} <-  muhat_{Sk}  k < j
                              muhat_{Sj} <-  tmp
                              muhat_{Sk} <-  muhat_{S(k+1)},  k>j, k<n.

                merge weights  w_{Sk} <- w_{sk}    k < j
                               w_{Sj} <- bar
                               w_{Sk} <- w_{s(k+1)}  k >j, k < n

Go to A.

Get final estimates by tracing back:
                               muhat.j <- muhat_{S}  if j in S
```

Here’s how it works on the toy example

```
          i   1    2    3    4    5    6    7    8
        y_i   0    1    1    0    0    1    1    1

        w_i   1    1    1    1    1    1    1    1

    y_i/w_i  0/1  1/1  1/1  0/1  0/1  1/1  1/1  1/1
                        ------ violators

             0/1  1/1     1/2    0/1  1/1  1/1  1/1
                  ---------- violators

             0/1      2/3        0/1  1/1  1/1  1/1
                      -------------- violators

             0/1            2/4       1/1  1/1  1/1  *no more violators*

    Traceback: muhat = (0, 1/2,1/2,1/2,1/2, 1,1,1 )
```

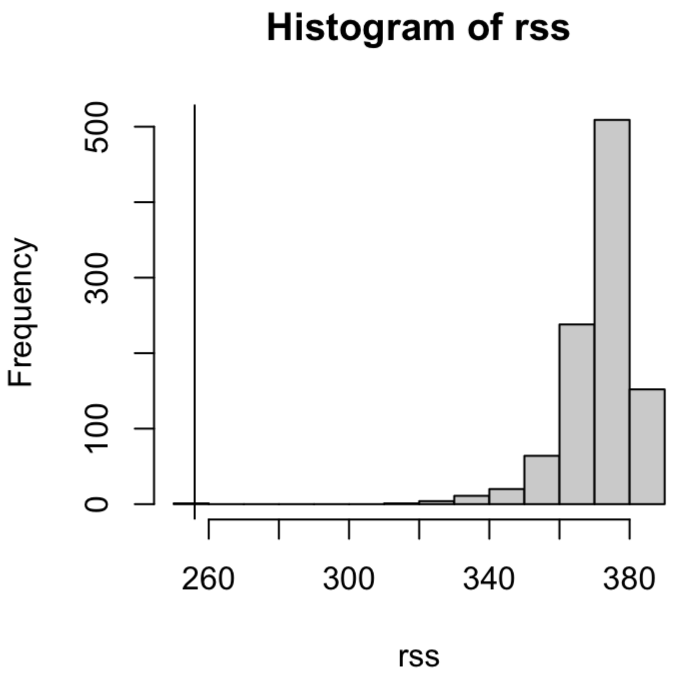
Example: Ice-on time

```
dat <- read.csv("MendotaIce.csv",header=TRUE)

# use isoreg, built-in PAVA-code for increasing isotonic regression;

x0 <- 2018 - dat$WINTER   # so we can consider non-decreasing
y0 <- dat$DAYS

ok <- !is.na(y0)   # clean data
x <- x0[ok] ; y <- y0[ok]
fit <- isoreg( x, y )

fit$x <- dat$WINTER[ok]
par( mar=c(3,3,3,0))
plot(fit, main="Ice-on days, Lake Mendota", xlab="year", ylab="days", las=1 )
legend( "bottomleft", legend="isotonic regression", lwd=1, col="red" )
```



It might be useful to check if the model fit is unusually good, compared to what one might expect if there is no change in expected value over time.

```
B <- 1000
rss <- numeric(B)
rss[1] <- var( fit$y[fit$ord] - fit$yf ) ## residual variance
n <- length(y)
for( b in 2:B )
 {
  fstar <- isoreg(x, y[sample(1:n)] )
  rss[b] <- var( fstar$y[fstar$ord]  - fstar$yf )
 }
hist( rss )
abline( v = rss[1] )
```



```
print( mean( rss <= rss[1] ) )
```

```
## [1] 0.001
```

Related work

Recent research provides a new view on isotonic regression. Instead of forcing strict, hard constraints on monotonicity, this work explores an approach to making a softer constraint. Connections to fused LASSO and data on word-wide temperature changes are discussed.

Tibshirani, Hofling, and Tibshirani, 2017 (https://www.researchgate.net/publication/229007529_Nearly-Isotonic_Regression)

The R function `isoreg` provides for standard isotonic regression. A more fully developed package `isotone` is available at CRAN. The authors provide a very sophisticated vignette on algorithmic approaches to the ordered means and related problems: de Leeuw, Hornik, and Mair (https://cran.r-project.org/web/packages/isotone/vignettes/isotone.pdf)

PAVA was first introduced by Ayer and colleagues in the context of the ordered-binomial-probabilities problem, which we'll discuss briefly next time: Ayer, Brunk, Ewing, Reid, and Silverman, 1955 (http://www.jstor.org/stable/2236377?seq=1#page_scan_tab_contents)

A useful textbook presentation is Groeneboom and Jongbloed, Nonparametric Estimation Under Shape Constraints (https://www.cambridge.org/core/books/nonparametric-estimation-under-shape-constraints/881B662EEF5B5266E5E4D80E6153FCDA)

Constraints and the SVD Factorization

Last week we saw that any $n \times p$ matrix X can be expressed $X = UDV^T$, for an $n \times p$ matrix U, a $p \times p$ matrix V, and a diagonal matrix D, where $U^TU = V^TV = I_p$ and where diagonal entries of D held the singular values, in decreasing order; and also in which exactly r of them are positive, with r the rank of X.

Our development of SVD began by constructing the right singular vectors (columns of V), the first of which we found by maximizing, over unit vectors, the Euclidean length $|Xv| = \sqrt{v^TX^TXv}$. We found subsequent right-singular vectors by maximizing over every more constrained sets of unit vectors; for instance, the 2nd right singular vector was obtained by maximizing that length over unit v perpendicular to the first right-singular vector.

With hindsight, and with tools of constrained optimization available, we have another approach to confirm properties of the right-singular vectors, Consider the differentiable *convex* function $f(v) = -|Xv|$, with gradient $\nabla f(v) = -(X^T X)v/|Xv|$.

The set of unit vectors in R^p is not convex, however, we can recognize that its convex hull is: introduce $C = \{v \in R^p : |v| \leq 1\}$. We may ask, then, what v minimizes $f(v)$ subject to $v \in C$. (We will find that to do so is maximize length, and so it provide a unit vector.) For a direct argument, suppose that X has the factorization above: $X = UDV^T$, with components as noted. Then look at first order optimality; doing so will confirm that the first column of V maximizes $|Xv|$ subject to the unit vector constraint. The interesting thing is that the exact same argument follows for the second, third, and remaining right-singular vectors, because the orthogonality constraints simply provide ever smaller convex constraint sets. This approach might be viewed as simpler than the previous induction argument, which tried to avoid the sequence of constraints. It is reassuring that different approaches are in alignment, and it is interesting that the tools of convex analysis provide a path to resolve complicated constrained optimizations.