CVX Implementation Notes

Michael J. Meyer

November 9, 2017

Contents

1 Basic terminology and setup

An *OptimizationProblem* consists of a convex *ObjectiveFunction* $f(x) \in \mathbb{R}$, $x \in \mathbb{R}^n$, a *ConstraintSet* (list of inequality constraints $g_j(x) \leq u_j$, $j = 1, \dots, m$) and possibly equality constraints in the form $Ax = b$, where A is an $n \times p$ matrix and $b \in \mathbb{R}^p$ a vector. This means that we have p equality constraints

$$row_k(A) \cdot x = b_k, \quad k = 1, \dots, p.$$

It is assumed that $p < n$ and that the matrix A is of full rank p (i.e. no equalities can be eliminated because they are merely a linear combination of other equalities). Typically p is much smaller than n.

The equality constraints are treated separately and are not included in the ConstraintSet of the problem. In other words: the ConstraintSet consists only of the inequality constraints.
The optimization problem now has the following form:

$$\text{Minimize } f(x) \quad \text{subject to } g_j(x) \leq u_j \text{ and } Ax = b. \tag{1}$$

The problem is called *feasible* if there exists a point $x \in \mathbb{R}^n$ satisfying all the constraints $g_j(x) \leq u_j$ and $Ax = b$ (a so called *feasible point*) and is called *infeasible* otherwise.

Clearly, if the problem is infeasible nothing further has to be done and some of the solution algorithms need a feasible point as a starting point.

Thus the first step in the solution must be to decide whether the problem is feasible at all and find a feasible point. In general this is a nontrivial problem requiring a *feasibility analysis* (think of thousands of variables x_j and hundreds of constraints).

Interestingly such an analysis can be carried out as a modified convex minimization problem

$$\text{Minimize } \tilde{f}(x, s) = s \quad \text{subject to } g_j(x) - s \leq u_j \text{ and } Ax = b. \tag{2}$$

for which a feasible point (x_0, s_0) is easily found (any solution $Ax_0 = b$ combined with sufficiently large s_0). If the minimization comes up with a point (x^*, s^*) satisfying $s^* \leq 0$, then x^* is a feasible point for the original problem. This is called *basic Phase I feasibility analysis.*

If the problem only admits feasible points x such that $g_j(x) = u_j$ for some j, then in general, in finite precision arithmetic, we cannot definitively decide if the problem is in fact feasible. Such a decision may still be possible in special cases (e.g. if the functions $g_j(x)$ are integer valued) but this will not be pursued here.

Our analysis will only be able to find feasible points x satisfying $g_j(x) < u_j - \epsilon$ or conclude that the problem is infeasible if a *certificate of infeasibility* can be found. To see what such a certificate might be let us assume that we have no equality constraints (the case with equality constraints is only slightly more involved) and consider the function

$$L(x, s, \nu) := \tilde{f}(x, s) + \sum\nolimits_{j=1}^{m} \nu_j (g_j(x) - s - u_j), \quad \nu \in \mathbb{R}^m.$$

If the vector ν satisfies $\nu_j \geq 0$, for all $j = 1, \dots, m$, then we have

$$L(x, s, \nu) \leq \tilde{f}(x, s) = s,$$

for all points (x, s) satisfying the constraints in (2). Taking the infimum over feasible points (x, s) we have

$$L_*(\nu) = \inf_{x,s} L(x, s, \nu) \leq \inf\{\, s \,:\, g_j(x) - s \leq u_j \,\}.$$

If it is known that $L_*(\nu) > 0$, then the constraints $g_j(x) \leq u_j$ are infeasible and the vector ν is a certificate of infeasibility. Some solution algorithms yield such ν as a byproduct of the computation.

Indeed, for any such ν the function $L(x, s, \nu)$ is convex in the variables (x, s) so that the equation

$$\nabla_{x,s} L(x_0, s_0, \nu) = 0$$

implies that the function $l(x, s) = L(x, s, \nu)$ assumes a global minimum at the point (x_0, s_0). In other words, we have $L_*(\nu) = L(x_0, s_0, \nu)$.

Some algorithms actually solve this equation for certain $\nu \in \mathbb{R}^m_+$ obtaining a solution (x_0, s_0) so that the value $L(x_0, s_0, \nu)$ itself provides the lower bound on the variable s in the problem (2).

Again it is clear that $L_*(\nu) = L(x_0, s_0, \nu) > 0$ can only be detected in finite precision arithmetic if in fact $L_*(\nu) \geq \epsilon > 0$, for some sufficiently large $\epsilon > 0$. Thus a grey zone remains in which the problem is feasible but this cannot be detected by our algorithms.

2 Handling the equality constraints

The equality constraints $Ax = b$ always form an underdetermined system. The solution set is an affine subspace of the space in which the decision variable x lives. We will deal with these constraints in one of two ways:

- Handle them in the KKT conditions, or
- Eliminate them by solving the system $Ax = b$ as $x = z + Fu$, where z is the minimum norm solution of $Ax = b$ and $Im(F) = ker(A)$, that is, $w = Fu$ is the general solution of $Ax = 0$. This approach will be called *reduction*.

Which of the two approaches is more efficient depends on the precise circumstances of the problem. For example in the Barrier method the inequality constraints are absorbed into the objective function leading to a new problem without inequality constraints. In this case the equality constraints increase the dimension of the resulting KKT system from $n \times n$ to $(n + p) \times (n + p)$.

On the other hand reduction is achieved by a change of variable $x \in \mathbb{R}^n \rightarrow u \in \mathbb{R}^{n-p}$ leading to a reduction in dimension. However this change of variables affects the computation of gradients and Hessian matrices:

$$\nabla_u h(z+Fu) = F'\nabla_x h(z+Fu) \quad \text{and} \tag{3}$$
$$\nabla_u^2 h(z+Fu) = F'\nabla_x^2 h(z+Fu)F \tag{4}$$

Here F is an $n \times p$-matrix, where usually p is much smaller than n. Consider the extreme case $p = 1$: in this case F is a large matrix and the matrix multiplications in the computation of $\nabla_u^2 h(z+Fu)$ are expensive.

If nothing is known about the structure of the function h this multiplication has to be carried out whenever the Hessian $\nabla_u^2 h(z+Fu)$ is evaluated at a new point u. In the course of the our minimization algorithms this has to done for the objective function $h = f$ as well as all the functions $h = g_j$ defining the inequality side conditions (which could number in the hundreds).

Clearly this has the potential to introduce catastrophic overhead. On the other hand, if h is known to be a quadratic function:

$$h(x) = r + a'x + \tfrac{1}{2}x'Px := h_{r,a,P}(x)$$

with a symmetric matrix P then, using that h agrees with its second order Taylor expansion and that $\nabla h(x) = a + Px$ and $\nabla^2 h(x) = P$, we get

$$\begin{aligned} h(z+Fu) &= h(z) + (Fu)'\nabla_x h(z) + \tfrac{1}{2}(Fu)'\nabla_x^2 h(z)Fu \\ &= h(z) + (Fu)'(a+Pz) + \tfrac{1}{2}u'(F'PF)u \\ &= h_{q,b,Q}(u), \end{aligned} \tag{5}$$

where

$$q = h(z),\ b = F'(a+Pz),\ \text{and } Q = F'PF.$$

In other words $h(z+Fu)$ is another quadratic function of u with known coefficients which can be precomputed. The change of variables $x \to u$ is then implemented simply by passing to the new coefficients which costs only one expensive matrix multiplication.

The gradient and Hessian with respect to the variable u can be read off from the new coefficients (at each point u) and incur no expensive matrix multiplications.

3 Regularization

The quadratic approximation of the objective function $f(x)$ at iterate x_k is

$$\tilde{f}(x_k + \Delta x) = f(x_k) + \nabla f(x_k)'\Delta x + \tfrac{1}{2}\Delta x' H_k \Delta x,$$

where H_k is the Hessian $\nabla^2 f(x_k)$ or an approximation thereof. The search direction Δx from iterate x_k is computed by minimizing this function over the variable Δx resulting in the equation

$$H_k \Delta x = -\nabla f(x_k) := -y_k \tag{6}$$

for the search direction Δx. Clearly we can assume that $y_k = \nabla f(x_k) \neq 0$ since the search terminates at a zero gradient .

In our algorithms we can also assume that H_k is positive semidefinite, but not necessarily nonsingular. If H_k is nonsingular (i.e. positive definite), then the solution Δx is a *descent direction*:

$$\Delta x' \nabla f(x_k) = -\Delta x' H_k \Delta x < 0$$

and this is all we care about. We will solve (22) by Cholesky factorization $H_k = LL'$ with lower triangular L, by solving the triangular systems

$$Lv = -y_k, \quad L'\Delta x = v. \tag{7}$$

The Cholesky factorization fails if H_k is singular. In that case we replace H_k with $H_k(\delta) := H_k + \delta I$, for some positive constant δ. This matrix is now positiv definite, in fact

$$(H_k(\delta)u, u) = u' H_k(\delta) u = u' H_k u + \delta u' I u \geq \delta \left\| u \right\|^2$$

so that (22) now yields a descent direction. Moreover it improves the conditioning of (22): if $H_k + \delta I = L(\delta)L(\delta)'$ is the Cholesky factoriztion of $H_k(\delta)$, then we have

$$|L(\delta)_{ii}| \geq \delta.$$

Indeed, the diagonal element $L(\delta)_{ii}$ is an eigenvalue of the triangular matrix $L(\delta)'$. Let u be a corresponding eigenvector. Then we have

$$|L(\delta)_{ii}|^2 \left\| u \right\|^2 = \left\| L(\delta)' u \right\|^2 = (L(\delta)L(\delta)' u, u) = ((H_k(\delta)u, u) \geq \delta \left\| u \right\|^2$$

from which it follows that

$$|L(\delta)_{ii}| \geq \sqrt{\delta}$$

with obvious implications for the numerical stability of the triangular systems (7). Moreover this suggests that we should replace H_k with $H_k + \delta I$ not only if the Cholesky factorization fails but rahter as soon as the minimal diagonal element (in absolute value) of the Cholesky factor L is below the threshold $\sqrt{\delta}$.

Trust region interpretation. The passage from the matrix H_k to the regularization $H_k + \delta I$ has an interpretation in terms of *trust regions*: the solution Δx^* of

$$H_k(\delta)\Delta x = -y_k, \qquad \text{where } y_k = \nabla f(x_k),$$

is the minimizer of the quadratic function

$$\begin{aligned}
\phi(\Delta x) &= f(x_k) + y_k' \Delta x + \Delta x' H_k(\delta) \Delta x \\
&= f(x_k) + y_k' \Delta x + \Delta x' H_k \Delta x + \delta \left\| \Delta x \right\|^2 \\
&= \tilde{f}(x + \Delta x) + \delta \left\| \Delta x \right\|^2 .
\end{aligned}$$

Now note that this minimizer Δx^* is automatically also the minimizer of the quadratic approximation $\tilde{f}(x_k + \Delta x)$ on the ball $B(x_k, r_k)$ with radius $r_k =$

$\|\Delta x^*\|$. Indeed, if this ball contained a point u with $\tilde{f}(x_k+u) < \tilde{f}(x_k+\Delta x^*)$, then, since also $\|u\| \leq \|\Delta x^*\|$ it follows that

$$\phi(u) = \tilde{f}(x_k+u) + \delta \|u\|^2 < \tilde{f}(x_k+\Delta x^*) + \delta \|\Delta x^*\|^2 = \phi(\Delta x^*).$$

In other words: passing from H_k to $H_k + \delta I$ we compute the search direction Δx by minimizing the quadratic approximation $\tilde{f}(x_k+\Delta x)$ not globally but instead on the ball $B(x_k, r_k)$ (the region in which we trust the approximation) where the trust radius r_k is defined implicitly as $r_k = \|\Delta x^*\|$.

This indicates that the regularization $H_k \to H_k(\delta)$ is not unreasonable and in any case it solves the problem of nonsingularity of H_k for us, improves the conditioning and results in a descent direction Δx at iterate x_k.

4 Hessian

Let $f : \mathbb{R}^n \to \mathbb{R}$ be a twice continuously differentiable function. The second order Taylor expansion of f centered at x has the form

$$f(x+h) = f(x) + L(h) + \tfrac{1}{2}B(h,h) + R(h),$$

where L is a linear function of $h \in \mathbb{R}^n$, $B(u,v)$ is a bilinear function of $(u,v) \in \mathbb{R}^n \times \mathbb{R}^n$ and the remainder $R(h)$ satisfies $R(h) = o(\|h\|^2)$. This condition on the remainder ensures that L and B are uniquely determined as

$$L(h) = \nabla f(x)'h \quad \text{and} \quad B(u,v) = u'Hv,$$

where $H := \nabla^2 f(x) \in Mat_{n\times n}(\mathbb{R})$ is the matrix with entries

$$H_{ij} = B(e_i, e_j) = \frac{\partial^2 f}{\partial x_i \partial x_j}(x),$$

i.e. the Hessian matrix of f at x.

To compute the gradient and Hessian of a C^2-function f we make use of the fact that the remaider condition $R(h) = o(\|h\|^2)$ in a quadratic expansion

$$f(x+h) = f(x) + \Delta'h + h'Hh + R(h)$$

uniquely determines the "coefficients" Δ and H as $\Delta = \nabla f(x)$ and $H = \nabla^2 f(x)$. We only need to find such an expansion for $f(x+h)$ and check that the remainder satisfies $R(h) = o(\|h\|^2)$. This is how we will derive our formulas below.

Hessian of affine transformation. Let $f : \mathbb{R}^n \to \mathbb{R}$ be a C^2-function and $\overline{f}(u) = f(x_0 + Fu)$, where $x_0 \in \mathbb{R}^n$ and $F : \mathbb{R}^m \to \mathbb{R}^n$ is a linear map, i.e. $F \in Mat_{n\times m}$.

We want to compute the gradient and Hessian of h at any point u from those of f. To get these do a second order Taylor expansion of f about x:

$$f(x+h) = f(x) + h^T g + h^T Hh + o(\|h\|^2),$$

where $g = \nabla f(x)$ and $H = \nabla^2 f(x)$ are uniquely determined by the fact that the residual is $o(\|h\|^2)$. Applying this to the point $x = x_0 + Fu$ this implies that

$$\begin{aligned}\overline{f}(u+h) &= f(x_0 + Fu + Fh)\\ &= f(x_0 + Fu) + (Fh)^T g + (Fh)^T HFh + o(\|Fh\|^2).\end{aligned}$$

Since $o(\|Fh\|^2)$ is $o(\|h\|^2)$ we conclude from this that

$$\nabla\overline{f}(u) = F^T g \quad \text{and} \quad \nabla^2\overline{f}(u) = F^T HF,$$

or, more explicitly

$$\nabla\overline{f}(u) = F^T \nabla f(x_0 + Fu) \quad \text{and} \quad \nabla^2\overline{f}(u) = F^T \nabla^2 f(x_0 + Fu)F. \tag{8}$$

Hessian of composition. With a similar approach we can compute the Hessian of a composition $g(f(x))$, where here $g : \mathbb{R} \to \mathbb{R}$ is a scalar function of one variable (more general g are much harder to handle and we do not need them). Indeed, set

$$y = f(x), \quad \nabla = \nabla f(x) \quad H = \nabla^2 f(x) \quad \text{and} \quad k = h^T\nabla + \tfrac{1}{2}h^T Hh$$

and use second order Taylorapproximations on f at the point x and g at the point $y = f(x)$ to obtain:

$$\begin{aligned}g(f(x+h))) &= g\left(f(x) + h^T\nabla + \tfrac{1}{2}h^T Hh\right)\\ &= g(y+k) = g(y) + g'(y) + \tfrac{1}{2}g''(y)k^2 + o(k^2)\\ &= g(y) + g'(y)\left(h^T\nabla + \tfrac{1}{2}h^T Hh\right) + \frac{1}{2}g''(y)\left(h^T\nabla + \tfrac{1}{2}h^T Hh\right)^2 + o(\|h\|^2).\end{aligned}$$

Here we have used that $o(k^2) = o(\|h\|^2)$. Collect terms of first and second order in h together and sticking all terms of higher order into the residual $o(\|h\|^2)$. Note that the squared term contributes no first order terms and only one second order term, this being the term

$$(h^T\nabla)^2 = (h^T\nabla)(h^T\nabla) = (h^T\nabla)(h^T\nabla)^T = h^T(\nabla\nabla^T)h.$$

We obtain

$$g(f(x+h))) = g(y) + h^T[g'(y)\nabla] + \frac{1}{2}h^T\left[g'(y)H + g''(y)\nabla\nabla^T\right]h + o(\|h\|^2).$$

and from this we can read off that

$$\nabla(g\circ f)(x) = g'(f(x))\nabla f(x), \quad \text{and} \tag{9}$$

$$H(g\circ f)(x) = g'(f(x))H + g''(f(x))\nabla f(x)\nabla f(x)^T \tag{10}$$

Note that here $d = \nabla f(x)$ is viewed as a column vector and so $\nabla f(x)\nabla f(x)^T$ is the *outer product*

$$\nabla f(x)\nabla f(x)^T = dd^T = (d_i d_j)_{ij}.$$

We will need the following example to construct test functions for unconstrained minimization:

Example 4.1. Let $\phi : G \subseteq \mathbb{R} \to \mathbb{R}$ be a function of one variable defined on an open subset $G \subseteq \mathbb{R}$, fix $a \in \mathbb{R}^n$ and set $g(x) = \phi(a \cdot x)$, for all $x \in \mathbb{R}^n$ such that $a \cdot x \in G$.

Since $f(x) = a \cdot x$ satisfies $\nabla f(x) = a$ and $H = \nabla^2 f(x) = 0$, for all $x \in \mathbb{R}^n$, the formulas (9) and (10) yield

$$\nabla g(x) = \phi'(a \cdot x)a \quad \text{and} \quad \nabla^2 g(x) = \phi''(a \cdot x)aa'.$$

The idea is to construct test functions of the form

$$objF(x) = \sum\nolimits_j \phi_j(a_j \cdot x)$$

where all the $\phi_j = \phi_j(u)$ have a unique minimum at $u = 0$. Then $objF(x)$ assumes its global minimum at all points x satisfying $a_j \cdot x = 0$, for all j, equivalently $Ax = 0$, where A is the matrix with rows a_j, provided such a solution exists.

If the functions ϕ_j are all convex, the same is true of our objective function $objF$ (sums and compositions of convex functions are again convex) With this we can construct examples with well conditioned, poorly conditioned and even singular Hessians ($ker(A) \neq \{0\}$) with known minimizers. The conditioning of $\nabla^2 f(x)$ is closely related to that of the matrix A.

5 Nullspace

Here we discuss methods to solve an underdetermined set of linear equations $Ax = b$ where the number n of variables is larger than the number m of equations. In other words A is an $m \times n$ matrix with $m < n$.

The set of solutions is then the hyperplane $x_0 + ker(A)$ where $ker(A)$ denotes the nullspace of A and x_0 is any particular solution of $Ax = b$.

We will represent the nullspace of A as the columnspace $colspace(F)$ of a matrix F such that

$$AF = 0.$$

If F satisfies this equation, then the column space of F is a subspace of the null space $ker(A)$. Recall also that the column space of F is the range $Im(F)$ of the linear map defined by the matrix F.

We will generally assume that A *has full rank*, i.e. $rank(A) = m$. The nullspace $ker(A)$ then has dimension $n - m$ and the columns of F span the entire nullspace of A if and only if $rank(F) = n - m$.

If we have found such a matrix F and x_0 with $Ax_0 = b$, then the hyperplane of all solutions to $Ax = b$ can be represented as

$$x_0 + ker(A) = x_0 + Im(F).$$

We want to apply this to the minimization problem

$$\text{minimize } f : \mathbb{R}^n \to \mathbb{R} \text{ under the constraint } Ax = b. \tag{11}$$

Since the solutions x to the constraint are exactly the vectors x of the form $x = x_0 + Fu$, $u \in \mathbb{R}^{n-m}$, we can turn this into the unconstrained problem

$$\text{minimize } g(u) = f(x_0 + Fu) \text{ on all of } \mathbb{R}^{n-m}. \tag{12}$$

Thereby we reduce the dimension of the problem from n to $n-m$. In large scale problems one would not do this since this operation can destroy the sparseness of the Hessian $H(f)$ which is critical for computation in very large dimensional problems.

In dense small problem (up to say $n = 3000$ variables) the above elimination of the constraint $Ax = b$ is a reasonable approach. We will discuss two algorithms to finding F and x_0. In both cases the columns of F will be orthonormal and hence an orthornomal basis for the null space $ker(A)$.

5.1 Null space with QR-decomposition of A

This approach relies on the relation $ker(A) = Im(A')^{\perp}$, where the prime denotes the matrix transpose and $V^{\perp}$ is the orthogonal complement of a subspace V. Recall that

$$\begin{aligned} n &= dim(ker(A)) + dim(ker(A)^{\perp}) = dim(ker(A)) + dim(Im(A')) \\ &= (n-m) + dim(Im(A')). \end{aligned}$$

Thus $dim(Im(A')) = m$. We will find the range $Im(A')$ and its orthogonal complement using the QR-decomposition of the transpose A' (which has dimension $n \times m$, i.e. maps from $\mathbb{R}^m$ to $\mathbb{R}^n$):

$$A' = QR, \tag{13}$$

where Q is orthogonal and R upper triangular with nonzero diagonal, where R is $p \times m$ (i.e. maps from $\mathbb{R}^m \to \mathbb{R}^p$ and Q is $m \times p$, i.e maps from $\mathbb{R}^p \to \mathbb{R}^m$.

In such a factorization Q and R must be $n \times p$ and $p \times m$ respectively, since A' is $n \times m$. Moreover $rank(Q) \geq rank(A') = rank(A) = m$ and so we must have $p \geq m$. Because the columns of Q are linearly independent and of dimension n we also must have $p \leq n$.

Indeed such factorizations exist for all $m \leq p \leq n$ but in practice (i.e. available in libraries) there are only two flavours:

(A) The reduced (minimal) form gives us R as an $m \times m$ matrix and $Q = [q_1, \dots, q_m]$ as an $n \times m$ matrix with m-columns $q_j \in \mathbb{R}^n$. It follows that R maps onto $\mathbb{R}^m = dom(Q)$ and hence

$$Im(A') = Im(Q) = span(q_1, \dots, q_m).$$

From this we see that $ker(A) = Im(A')^{\perp} = span(q_1, \dots, q_m)^{\perp}$ but if we use the reduced form we have to compute this orthogonal complement ourselves. In other words we have to extend $\{ q_1, \dots, q_m \}$ to an orthonormal basis $\{ q_1, \dots, q_m, q_{m+1}, \dots q_n \}$ of $\mathbb{R}^n$ ourselves.

In R the minimal form is obtained as follows

```
qrA <- qr(t(A)); Q <- qr.Q(qrA); R <- qr.R(qrA)
```

Here we first compute a QR-decomposition object qrA and from this object extract the matrices Q and R using the helper functions qr.Q and qr.R.

The intermediate forms of the decomposition (13) with $m \leq p \leq n$ extend this matrix Q by adding (arbitrarily) orthonormal columns on the right

$$Q \to Q_+ = [q_1, \dots, q_m, q_{m+1}, \dots q_p]$$

and adjusting the matrix R by adding zero rows at the bottom (so that R becomes $p \times m$. This of course does not change the matrix product QR as we can see from block multiplication

$$Q_+ = [Q, Q_1],\ R_+ = \begin{pmatrix} R \\ 0 \end{pmatrix} \quad \Longrightarrow \quad Q_+R_+ = QR + Q1 * 0 = QR.$$

(B) The full (maximal) form of the decomposition has $p = n$, that is, the columns of Q have been extended to a full orthonormal basis of $\mathbb{R}^n$ and so clearly

$$ker(A) = Im(A')^{\perp} = span(q_1, \dots, q_m)^{\perp} = span(q_{m+1}, \dots, q_n). \tag{14}$$

In other words the matrix F can be chosen to be the matrix with columns $q_{m+1}, \dots, q_n$:

$$F = [q_{m+1}, \dots, q_n] \in Mat_{n\times(n-m)}(\mathbb{R}). \tag{15}$$

To get a special solution x_0 of $Ax = b$ note that $A = R'Q'$ and solve the factored form $R'Q'x = b$. Set $y = Q'x$. Then *forward solve* the lower triangular system $R'y = b$ and get x from $Q'x = y$ as $x = Qy$.

To get the complete factorization $A' = Q_+R_+$ in R we do

```
qrA <- qr(t(A)); Qp <- qr.Q(qrA,complete=TRUE); R <- qr.R(qrA)
```

and then extract the matrix Q via `Q <- Qp[,1:m]`. R is already in incomplete form (no zero rows at bottom) since we have not specified complete=TRUE in the function qr.R (the default is complete=FALSE).

With this the factorization of A' becomes $A' = QR$ (and not $A' = QpR$) and this is what we need in the computation of the special solution x_0 above.

The same holds true in the scala breeze library where we get the QR decomposition as

```
val qr.QR(q,r)=qr(A).
```

This yields $Q = q$ as $n \times n$-matrix but yields $R = r$ as $m \times m$-matrix, that is, without the $n - m$ bottom zero rows. With this the decomposition of A' becomes $A' = Q[\cdot, 1:m] * R$ and so $A = R'P'$ where the matrix

$$P = Q[\cdot, 1:m] = [col_1(Q), \dots, col_m(Q)]$$

has orthogonal columns and we must solve for x_0 as $R'y = b$, $P'x = y$, that is, $x = Py$. Let us note that this x_0 is the minimal norm solution of the system $Ax = b$. Indeed $x_0 = Py \in span(col_1(Q), \dots, col_m(Q))$ while $Fu \in span(col_{m+1}(Q), \dots, col_n(Q))$ and so $Fu \perp x_0$, for all $u \in \mathbb{R}^{n-m}$, from which the claim follows.

The QR factorization is computed by repeatedly applying (orthogonal) Householder updates and is thus a very stable algorithm.

5.2 Nullspace via SVD decomposition of A

We can also compute the nullspace $ker(A)$ and special solution x_0 of $Ax = b$ using the more involved *SVD-decomposition* of A:

$$A = U\Sigma V'$$

where $U \in Mat_{m\times m}(\mathbb{R})$ and $V \in Mat_{n\times m}(\mathbb{R})$ are orthonormal matrices and Σ is an $m \times n$ diagonal matrix with entries

$$\sigma_1 \geq \sigma_2 \cdots \geq \sigma_m \geq 0$$

(the singular values of A). Since $rank(A) = m$ we must have $\sigma_i > 0$ and so the matrix Σ is invertible. Write $V = [v_1, \dots, v_m]$, where the $v_j \in \mathbb{R}^n$ are the columns of V. Thus, for all $x \in \mathbb{R}^n$,

$$\begin{aligned} Ax = 0 \quad &\Longleftrightarrow \quad U\Sigma V'x = 0 \\ &\Longleftrightarrow \quad \Sigma V'x = 0 \\ &\Longleftrightarrow \quad 0 = V'x = (v_1 \cdot x, v_2 \cdot x, \dots, v_m \cdot x)' = 0 \\ &\Longleftrightarrow \quad x \perp \{\, v_1, \dots, v_m \,\} \\ &\Longleftrightarrow \quad x \in span(v_{m+1}, \dots, v_n) \end{aligned}$$

In other words

$$ker(A) = span(v_{m+1}, \dots, v_n). \tag{16}$$

where $v_{m+1}, \dots, v_n \in \mathbb{R}^n$ are vectors which extend $v_1, \dots, v_m$ to an ON-basis of $\mathbb{R}^n$. To get the full $n \times n$ matrix

$$V_+ = [V, v_{m+1}, \dots, v_n]$$

we must compute the SVD in `R` via

```
svdA <- svd(A,nv=n); Vp <- svdA$v; V <- V[,1:m]
```

Now a particular solution of $U\Sigma V'x = Ax = b$ can be found by solving $\Sigma V'x = U'b := c$ which is equivalent to

$$V'x = (c_1/\sigma_1, \dots, c_m/\sigma_m)',$$

A particular solution of this is given by

$$x_0 = Vw, \quad \text{where} \quad w = (c_1/\sigma_1, \dots, c_m/\sigma_m)' \in \mathbb{R}^m. \tag{17}$$

since the columns of V are orthonormal.

6 Solving a system of equations with the SVD

The SVD decomposition is a convenient way to deal with singular systems of linear equations. Consider the system $Ax = b$, where A is any given $n \times n$ matrix and b and $n \times 1$ vector. Let

$$A = UDV' \tag{18}$$

be the SVD of A. Here D is an $r \times r$ diagonal matrix, where r is the rank of A and the diagonal elements σ_j are the *nonzero* singular values of A, in particular $\sigma_j > 0$.

U and V are $n \times r$ matrices with orthonormal columns: $U'U = I_{r\times r} = V'V$. Note that this *does not imply* that $UU' = I_{n\times n}$ or $VV' = I_{n\times n}$ since $r < n$ (for reasons of rank). Indeed these equalities hold exactly if $r = n$.

Write $U = (u_1, v_2 \dots, u_r)$ and $V = (v_1, v_2, \dots, v_r)$, where the u_j and v_j are the columns of U and V respectively. Then the decomposition 18 has the form

$$A = (u_1, v_2 \dots, u_r) \begin{pmatrix} \sigma_1 & 0 & \dots & 0 \\ 0 & \sigma_2 & \dots & 0 \\ \vdots & \vdots & \dots & \vdots \\ 0 & 0 & \dots & \sigma_r \end{pmatrix} \begin{pmatrix} v_1' \\ v_2' \\ \vdots \\ v_r' \end{pmatrix} = \sum\nolimits_{j=1}^{r} \sigma_j u_j v_j' \tag{19}$$

with strictly positive σ_j. It follows that A is the operator

$$Ax = \sum\nolimits_j \sigma_j (v_j' x) u_j = \sum\nolimits_j \sigma_j (x, v_j) u_j \tag{20}$$

with linearly independent coefficient functionals $\sigma_j(\cdot, v_j)$. It follows immediately that

$$Im(A) = Im(U) = span(\{u_1, u_2, \dots, u_r\}).$$

especially $rank(A) = r$ so that A is nonsingular if and only if $r = n$. Moreover

$$ker(A) = \{v_1, \dots, v_r\}^{\perp} = ker(V'). \tag{21}$$

As an aside let us note that $P = UU'$ is the orthogonal projection onto the range of U: indeed $P^2 = (UU')(UU') = U(U'U)U' = UU' = P$ and P is self adjoint and hence is the orthogonal projection onto its range. It remains to be seen that $Im(P) = Im(U)$. Clearly $Im(P) = Im(UU') \subseteq Im(U)$. Conversely let $y \in Im(U)$, say $y = Ux$. Then $y = U(U'U)x = (UU')Ux \in Im(UU')$. Likewise VV' is the orthogonal projection onto the range of V.

Note that $Im(U) \subseteq \mathbb{R}^n$ has dimension $r = rank(U)$ which may be less than n and this is the case if and only if the matrix A is singular.

From $A = UDV'$ it follows directly that $Im(A) \subseteq Im(U)$. On the other hand $AV = UD$ so that $Im(A) \supseteq Im(UD) = Im(U)$.

Since $Im(A) = Im(U)$ in general only the system $Ax = Pb$ is solvable and any solution is a minimizer of the norm $\|Ax - b\|$. Now we will solve this system using the representation (20).

Rewrite $Ax = Pb$ as

$$\sum\nolimits_{j=1}^{r} \sigma_j (v_j' x) u_j = \sum\nolimits_{j=1}^{r} (u_j' x) u_j \tag{22}$$

We get the solution by comparison of coefficients in the expansion in terms of the orthonormal u_j. Enlarge $\{v_1, \dots, v_r\}$ to an orthonormal basis $\{v_1, \dots, v_n\}$ of $\mathbb{R}^n$. Then x has the form

$$x = \sum\nolimits_{j=1}^{n} (v_j' x) v_j \tag{23}$$

but the left hand side of (22) does not depend on the coefficients $v_j' x$, where $j > r$, so that these can be arbitrary, while for $j < r$ we want $\sigma_j (v_j' x) = u_j' b$, that is,

$$v_j' x = \frac{u_j' b}{\sigma_j}.$$

Putting this into (23) we see that the general solution of $Ax = Pb$ is given by

$$x = \sum\nolimits_{j \le r} \frac{u_j' b}{\sigma_j} v_j + \sum\nolimits_{j=r+1}^{n} \lambda_j v_j \tag{24}$$

with arbitrary scalars λ_j. Obviously then the solution of minimal norm to the equation $Ax = Pb$ is given by

$$x = \sum\nolimits_{j \le r} \frac{u_j' b}{\sigma_j} u_j. \tag{25}$$

This x is then the minimizer of $\|Ax - b\|$ which has itself minimal norm. Note that we can directly compute the error $\|Ax - b\|$ as follows

$$\|Ax - b\| = \left\| b - \sum\nolimits_{\sigma_j > 0} (u_j' b) u_j \right\|. \tag{26}$$

Now minimization of the norm $\|Ax - b\|$ leads to a solution of the system $Ax = b$ if and only if this system has a solution and in this case (25) is the solution of minimal norm. Let us verify that (25) is indeed a solution of

$$Ax = Pb = \sum\nolimits_{j \le r} (u_j' b) u_j.$$

Because of orthonormality of the v_j we have $V' v_j = 0$, for $j > r$, and $V' v_j = e_j$ (the standard vector), for $j \le r$. Thus

$$V' x = \sum\nolimits_{j \le r} \frac{u_j' b}{\sigma_j} e_j.$$

The claim now follows if we observe that $De_j = \sigma_j e_j$ and $Ue_j = u_j$.

Software packages will often provide the *full* SVD (18) where U, V, D are all $n \times n$ and D has zeros on the main diagonal (the zero singular values of A). This can be obtained from the above reduced SVD as follows: extend the columns of U and V to full orthonormal bases

$$\{u_1, \ldots, u_r, u_{r+1}, \ldots, u_n\} \quad \text{and} \quad \{v_1, \ldots, v_r, v_{r+1}, \ldots, v_n\},$$

write $\sigma_j = 0$, for $j = r+1, \ldots, n$, and let $\overline{U}$, $\overline{V}$ and $\overline{D}$ be the matrices

$$\overline{U} = (u_1, \ldots, u_n), \quad \overline{V} = (v_1, \ldots, v_n), \quad \text{and} \quad \overline{D} = diag(\sigma_1, \ldots, \sigma_n)$$

i.e. in block matrix form

$$\overline{U} = (U, U_c), \quad \overline{V} = (V, V_c) \quad \text{and} \quad \overline{D} = \begin{pmatrix} D & 0 \\ 0 & 0 \end{pmatrix},$$

where $U_c = (u_{r+1}, \ldots, u_n)$, $V_c = (v_{r+1}, \ldots, v_n)$ and "0" denotes an $(n-r) \times (n-r)$ bock of zeros. With this we have an analogous decomposition

$$\overline{UDV}' = (U, U_c) \begin{pmatrix} D & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} V' \\ V_c' \end{pmatrix} = (U, U_c) \begin{pmatrix} DV' \\ 0 \end{pmatrix} = UDV' = A.$$

From now on we will write U, V, D also for the full matrices $\overline{U}, \overline{V}, \overline{D}$ since it will be clear from the context whether the full or reduced matrices are considered.

Then we have $U'U = UU' = I_{n \times n} = V'V = VV'$ and we will now find necessary and sufficient conditions for the system $Ax = b$ to have a solution. Rewrite this system as $UDV'x = b$, equivalently $Uz = b$, where $z = DV'x$.

Here U is $n \times n$ orthogonal so that the first equation always has a solution $z = U'b$. The second equation $DV'x = z$ has a solution exactly if the diagonal elements σ_j of D satisfy

$$\sigma_j = 0 \quad \Longrightarrow \quad z_j = u_j'b = 0.$$

In this case the general solution is given by

$$x = \sum\nolimits_j \lambda_j v_j, \quad \text{where} \quad \lambda_j = \begin{cases} \frac{u_j'b}{\sigma_j} & \text{if } \sigma_j > 0 \\ \text{arbitrary} & \text{if } \sigma_j = 0. \end{cases} \tag{27}$$

7 Solving the KKT system

The BarrierSolver absorbs the inequality constraints into the objective function so that only the equality constraints are left in the KKT conditions. Thus we deal with this case first.

7.1 KKT system without inequality constraints

If there are no inequality constraints, the KKT system has the form

$$\begin{aligned} Hx + A'\nu &= -q \qquad (28)\\ Ax &= b \qquad (29)\end{aligned}$$

Here H is symmetric and positive semidefinite (convexity!) and A satisfies $ncol(A) = ncol(H)$. First we assume that H is balanced and positive definite. Then we proceed as follows: do a Cholesky factorization $H = LL'$. Solve

$$HX = LL'X = [A', q] \quad \text{via} \quad LY = [A', q] \text{ then } L'X = Y.$$

Obtain $H^{-1}A'$ and $H^{-1}q$. From the first equation get $x = -H^{-1}(q + A'\nu)$. Stick this into the second equation to obtain

$$AH^{-1}A'\nu = -(b + AH^{-1}q).$$

Here the matrix $S = AH^{-1}A'$ is symmetric and positive semidefinite. It is also invertible since A has full rank. Thus it is positive definite and can be solved with a Cholesky factorization

$$S = R'R.$$

Moreover this matrix is small ($p \times p$) and so this effort is negligible. We set $\mu = R\nu$ and solve for ν in 2 steps via

$$R'\mu = -(b + AH^{-1}q) \text{ then } R\nu = \mu.$$

Next we treat the general case. First we equilibrate the matrix H by passing to $H \to DHD$, for a suitable diagonal matrix $D = diag(d)$ with $d_i \neq 0$, for all i. Setting $x = Dy$ we can rewrite (28) as

$$\begin{aligned} HDy + A'\nu &= -q\\ ADy &= b\end{aligned}$$

Multiply the first equation with D on the left to get

$$\begin{aligned} My + B'\nu &= -Dq \qquad (30)\\ By &= b \qquad (31)\end{aligned}$$

with $M = DHD$ and $B = AD$. This system is of the same form but with a balanced matrix M instead of H. I.e. we apply the transformation

$$x \to y = D^{-1}x, \quad H \to DHD, \quad A \to AD, \quad q \to Dq. \qquad (32)$$

After solving this new system as outlined above we have to undo the transformation $x \to D^{-1}x$ by replacing $x \to Dx$.

If now DHD is not positive definite (this happens exactly if H is not positive definite since the d_i are all nonzero), we pass to the equivalent system

$$DHD \to DHD + rA'A, \quad -q \to -q + rA'b. \tag{33}$$

Here $DHD + rA'A$ is positive definite if and only if the KKT system (30) itself is nonsingular. If it is singular, we stop. Assume now it is nonsingular. Then the new matrix H is positive definite.

7.2 Singular KKT systems

Singular Hessians H can occur naturally in the barrier function of a phase I feasibility analysis if the inequality constraints do not depend on some of the variables, i.e. if some variables are unconstrained or constrained only by equality constraints (the corresponding row of the Hessian is then zero).

Then a Cholesky factorization of H fails and the solution by block elimination is not possible. If there are no equality constraints the KKT system itself has zero rows and so is singular but it still may have solutions. The best way to deal with this situation is to move on to the more expensive solution via the SVD decomposition of the full KKT matrix.

8 Regularization of ill conditioned systems

This section is a very brief and incomplete summary of [1]. In case the system $Ax = b$ is ill conditioned, the least squares solution $x_0 = (A^*A)^{-1}Ab$ computed from the normal equations is mostly useless. Part of the problem is that the matrix A^*A is even more ill conditioned than the matrix A itself.

The problem can be seen more explicitly as follows: let $A = UDV'$ be the SVD of A where $D = diag(\sigma_j)$ is a diagonal matrix with the singular values of A on the main diagonal. Then the least squares solution x_0 of minimal norm, i.e. the minimizer of $\|Ax - b\|$ which has itself minimal norm is given by the expression

$$x_0 = \sum\nolimits_{\sigma_j > 0} \frac{u_j'b}{\sigma_j} v_j.$$

If now b has a significant component $u_j'b$ in the direction of some u_j where the singular value σ_j is very small then we are dividing a number of significant size by a very small number which can lead to error blowup.

For this reason we have to modify the matrix A to avoid small singular values. The basic ill conditioned problem is numeric differentiation and this can be used to motivate the following approach: the system $Ax = b$ is reformulated as

$$Ax = b \quad \text{subject to } x = Sw, \tag{34}$$

where S is a "smoothing operator". I.e. we enforce some smoothness on the solution x by requiring it to be in the range of S. The basic example, numeric differentiation ([1], p5, eq(1)), suggests the operator

$$S = \begin{cases} (A^*A)^{p/2} & \text{if } p \text{ is odd,} \\ (A^*A)^{(p-1)/2}A^* & \text{if } p \text{ is even,} \end{cases} \tag{35}$$

which in this example is p-fold iterated integration. In general the integer p is a parameter that controls the smoothness of the solution x.

Note that the range of S shrinks as the parameter p increases. Since (34) stipulates that the solution x must come from the range of S this condition becomes more and more strict as p increases.

In the basic example of numeric differentiation the parameter p is literally the degree of differentiability. This parameter has to be estimated. Let us note that

$$SS^* = (A^*A)^p, \quad \forall p. \tag{36}$$

Regularization of the system $Ax = b$ replaces the inverse A^{-1} by a family C_h, $h > 0$, of matrices such that

$$C_hA \to I, \quad \text{as } h \downarrow 0,$$

where I denotes the identity matrix as usual. The C_h are chosen such that the constants

$$\gamma_1 = \sup_{h>0} h \|C_h\| \quad \text{and} \quad \gamma_2 = \sup_{h>0} h^{-p} \|(I - C_hA)S\|, \tag{37}$$

where p is as above, are finite and not too large. Then, by definition,

$$\|C_h\| \le \gamma_1/h \quad \text{and} \quad \|(I - C_hA)S\| \le \gamma_2 h^p. \tag{38}$$

This implies the following fundamental error bound:

Proposition 8.1. *If $\Delta > 0$, $x = Sw$ and $\|Ax - b\| \le \Delta \|w\|$, then we have*

$$\|x - C_hb\| \le ((\gamma_1\Delta)/h + \gamma_2 h^p) \|w\|. \tag{39}$$

In other words: if the problem $Ax = b$ with $x = Sw$ is approximately solvable at all to within error $\Delta \|w\|$, then $x_0 := C_hy$ will differ from any such approximate solution x by at most the right hand side of (39), meaning of course that we will use x_0 as a practical solution of (34).

Proof.

$$\begin{aligned} \|x - C_hb\| &= \|(x - C_hAx) + (C_hAx - C_hb)\| \\ &= \|(I - C_hA)Sw + C_h(Ax - b)\| \\ &\le \|(I - C_hA)S\| \|w\| + \|C_h\| \|Ax - b\| \\ &\le \|(I - C_hA)S\| \|w\| + \Delta \|C_h\| \|w\| \end{aligned}$$

and the claim follows by substituting the upper bounds from (38) for $\|(I - C_h A)S\|$ and $\|C_h\|$. ▮

Rewriting the system $Ax = b$ as in (34) is an *assumption* that we can solve the system with a smooth solution $x = Sw$ while the bounds (38) can be satisfied by suitable choice of the approximate inverses C_h of A.

The optimal parameter $h > 0$ for the (approximate) solution $x_0 = C_h b$ is then computed by minimizing the right hand side

$$RHS(h) := (\gamma_1 \Delta)/h + \gamma_2 h^p$$

of (39). Setting the derivative $RHS'(h)$ equal to zero immediately yields the following equality for the optimal h

$$p\gamma_2 h^{p+1} = \gamma_1 \Delta \tag{40}$$

that is, explicitly,

$$h = \left(\frac{\gamma_1 \Delta}{p\gamma_2}\right)^{1/(p+1)}.$$

Using (40) to simplify $RHS(h)$ for the optimal h we obtain

$$RHS(h) = \gamma_2 (p+1) h^p$$

so that (39) simplifies to

$$\|x - C_h b\| \leq \gamma_2 (p+1) h^p \, \|w\| = \kappa \Delta^{p/(p+1)} = O(\Delta^{p/(p+1)}) \tag{41}$$

with a constant κ given by

$$\kappa = \|w\| \, \gamma_2 (p+1) \left(\frac{\gamma_1}{p\gamma_2}\right)^{p/(p+1)}.$$

Here we are interested in the behavior as $\Delta \to 0$. Note that an exact solution x of the system $Ax = b$ with $x = Sw$ satisfies the assumptions of the above proposition for all $\Delta > 0$ but the discussion applies also to the case where this system does not have an exact solution.

To construct suitable approximate inverses C_h we use the following proposition:

Proposition 8.2. *Set $Q = A^*A$, let S be any (smoothing) matrix satisfying $SS^* = (A^*A)^p = Q^p$ and $\phi : [0, +\infty) \to \mathbb{R}$ any continuous function satisfying*

$$\gamma_1 := \sup\nolimits_{t>0} |\phi(t)| t^{1/2} < +\infty \quad \textit{and} \quad \gamma_2 := \sup\nolimits_{t>0} |1 - t\phi(t)| t^{p/2} < +\infty \tag{42}$$

Then the family

$$C_h := h^{-2} \phi(h^{-2} Q) A^*, \quad h > 0, \tag{43}$$

satisfies the conditions (37) with the very same constants γ_1 and γ_2 (for the operator norm with respect to the L^2-norm).

Proof. The square of the operator norm of a matrix B with respect to the L^2-norm is the largest eigenvalue of BB^*. Note that $\phi(h^{-2}Q)$ is self-adjoint and that

$$C_h A = h^{-2}\phi(h^{-2}A^*A)A^*A = h^{-2}\phi(h^{-2}Q)Q.$$

Set $R = I - C_h A = I - h^{-2}\phi(h^{-2}Q)Q$. Note that R is self-adjoint. With this we obtain $\|C_h\|^2$ and $\|(I - C_h A)S\| = \|RS\|$ as the largest eigenvalue of

$$\begin{aligned} C_h C_h^* &= h^{-4}\phi(h^{-2}Q)A^*A\phi(h^{-2}Q) = h^{-4}Q\phi^2(h^{-2}Q) \\ &= \psi_h(Q), \end{aligned}$$

where $\psi_h(u) = h^{-4}u\phi^2(h^{-2}u)$, respectively

$$\begin{aligned} (RS)(RS)^* &= R(SS^*)R = RQR \\ &= [I - h^{-2}\phi(h^{-2}Q)Q]Q[I - h^{-2}\phi(h^{-2}Q)Q] \\ &= \chi_h(Q), \end{aligned}$$

where $\chi_h(u) = [1 - h^{-2}u\phi(h^{-2}u)]^2 u$. Let λ_i denote the eigenvalues of $Q = A^*A$. Then the eigenvalues of $C_h C_h^*$ are the numbers $\psi_h(\lambda_i)$ and the eigenvalues of $(RS)^*(RS)$ are the numbers $\chi_h(\lambda_i)$. The claim follows if we can show that

$$\sup_h \sup_i \psi_h(\lambda_i) < \infty \quad \text{and} \quad \sup_h \sup_i \chi_h(\lambda_i) < \infty.$$

Since there are only finitely many λ_i it will suffice to show that

$$\sup_h \psi_h(u) < \infty \quad \text{and} \quad \sup_h \chi_h(u) < \infty$$

for each $u > 0$ and this follows from the assumptions on the function ϕ. ∎

Example 8.1. Tichonov regularization Here $\phi(u) = 1/(u + 1)$ and

$$C_h = h^{-2}(I + h^{-2}A^*A)^{-1}A^* = (A^*A + h^{-2}I)^{-1}A^*.$$

For this ϕ one computes the constants γ_i from proposition 8.2 as $\gamma_1 = 1/2$ and

$$\gamma_2 = \begin{cases} 1/2 & \text{if } p = 1, \\ 1 & \text{if } p = 2 \quad \text{and} \\ +\infty & \text{if } p > 2. \end{cases}$$

In particular Tichonov regularization cannot take advantage of smoothness of degree $p > 2$.

Example 8.2. Iterated Tichonov regularization Let $l > 0$ be an integer. Then the function

$$\phi_l(u) = t^{-1}(1 - (1 + t)^{-l}$$

Corresponds to l-times iterated Tichonov regularization ([1], p10, proposition 5.2). It has finite γ-constants up to smoothness of degree $p = 2l$.

8.1 Using the SVD of A

Using the SVD decomposition $A = UDV'$, where $D = diag(\sigma_j)$ is the diagonal matrix containing the singular values of A, we can compute the matrix function $\phi(h^{-2}Q)$ in explicit form.

Note first that $Q = A^*A = VDU'UDV' = VD^2V'$ since the orthogonal matrix U satisfies $U'U = I$. From this it follows that $Q^k = V(D^2)^kV'$ for each integer power $k \geq 0$, whence $p(Q) = Vp(D^2)V'$ for each polynomial $p = p(u)$ where p acts on the diagonal elements of the diagonal matrix D^2, that is,

$$p(D^2) = diag(p(\sigma_j^2)).$$

Approximation by polynomials then yields

$$\phi(Q) = V\phi(D^2)V' = Vdiag(\phi(\sigma_j^2))V' \tag{44}$$

for each continuous function $\phi : [0, +\infty) \to \mathbb{R}$. Because of the compactness of the spectrum we need to approximate only on compact sets. In particular

$$\phi(h^{-2}Q) = V\phi(D^2)V' = Vdiag(\phi(\sigma_j^2/h^2))V', \quad h > 0 \tag{45}$$

(replace $\phi(u)$ with $\phi(h^{-2}u)$ for fixed $h > 0$).

8.2 Stochastic formulation

The constants γ_1 and γ_2 can be computed directly from our choice of the function ϕ in the construction of the approximate inverses C_h of A. However we also need the smoothness paramter p and given p, the optimal parameter value h from (40) and for this we need the parameter Δ.

Δ turns out to be hard to estimate in general so we turn to a reformulation of the problem in stochastic terms. We write

$$Ax = b + \epsilon \quad \text{and} \quad x = Sw, \tag{46}$$

where now ϵ and w are *mean zero* random vectors with pairwise uncorrelated components

$$E(\epsilon w') = 0, \quad E(\epsilon\epsilon') = \sigma^2 I, \quad \text{and} \quad E(ww') = \tau^2 I.$$

We then set

$$\Delta = \sigma/\tau$$

for the level or relative noise (input / output) of the system. Here the assumption of zero mean for the variable w is a bit jarring but needed to support the following proceedings. This quantity Δ is not related to the quantity Δ in the preceding discussion.

With these assumptions one can show ([1], theorem 8.1, p14) that the expected error $E\,\|x - Cy\|$ is minimized for the choice of approximate inverse C of A given by

$$C := (SS^*A^*A + \Delta^2 I)^{-1}SS^*A^* \tag{47}$$

with no assumptions on the smoothing matrix S. For this C we can compute the optimal estimator $x = Cb$ as follows: setting $P = AS$ we have

$$x = Sw, \quad \text{with} \quad (P * P + \Delta^2 I)w = P^* b. \tag{48}$$

This still leaves open the question how the new parameter Δ should be estimated form the right hand side y but this is now a simpler problem.

Typically the noise level σ in the right hand side b will be small while the noise level τ for w will be large (recall w centered at zero) so that the quotient $\Delta = \sigma/\tau$ is small. This is desirable so that the regularization term $\Delta^2 I$ does not dominate the computation.

Here the degree of smoothness is implicit in the smoothness matrix S. In the particular case of S as in (35) it is given by the parameter p and should be estimated rather than chosen arbitrarily.

For this particular smoothing matrix S, using the SVD decomposition $A = UDV'$ with $D = diag(\sigma_j)$ the computation of the optimal estimator $x = Cb$ in (48) can be simplified as follows:

$$x = Vz \quad \text{where } z_j = \frac{\sigma_j^{2p+1} c_j}{\sigma_j^{2p+2} + \Delta^2} \tag{49}$$

where the vector $c = (c_j)$ is given by $c = U'b$. In other words we have

$$x = \sum\nolimits_j \frac{\sigma_j^{2p+1} u_j' b}{\sigma_j^{2p+2} + \Delta^2} v_j \tag{50}$$

where $u_j = col_j(U)$ and $v_j = col_j(V)$. Clearly we need to sum only over the nonzero singular values $\sigma_j > 0$.

Estimation of Δ.

Let c be the vector $c = U'b$ above. From $b = Ax + \epsilon = ASw + \epsilon$ we see that b has mean zero and from the assumptions on the correlations between the ϵ_i and w_j we get

$$Cov(b) = E(bb') = E[(ASw + \epsilon)(w'S'A' + \epsilon')] = \tau^2 ASS'A' + \sigma^2 I. \tag{51}$$

Thus $c = U'b$ also has mean zero and covariance matrix

$$Cov(c) = E(cc') = U'Cov(b)U.$$

Observing that $U'U = I$, $V'V = I$ and recalling that $SS' = (A'A)^p$ and $AA' = UDV'VDU' = UD^2U'$ we have

$$ASS'A' = (AA')^{p+1} = (UD^2U')^{p+1} = UD^{2p+2}U'$$

so that $U'(ASS'A')U$ is the diagonal matrix D^{2p+2}. Now it follows from (51) that

$$Cov(c) = U'Cov(b)U = \tau^2 D^{2p+2} + \sigma^2 I \tag{52}$$

and in particular

$$E(c_j^2) = \tau^2\sigma_j^{2p+2} + \sigma^2 \tag{53}$$

and we have one realization of the (many) variables c_j but only two unknown parameters τ and σ to be estimated.

8.3 Parameter estimation

Now we turn to the following problem ([1], p18, section 10): We have *a single* realization of a high dimensional random vector $X = (X_1, \dots, X_n)$ known to satisfy

$$EX = v(\theta), \quad \text{that is, } EX_j = v_j(\theta), \tag{54}$$

where $v(\theta) = (v_1(\theta), \dots, v_n(\theta))$ is a function of a low dimensional parameter θ. We need to estimate the parameter θ from the one realization of X which we have.

In this situation we have only a few unknown values θ_k and many equations $EX_j = v_j(\theta)$ to work with but have only one realization of X to provide information about the mean EX. For this problem Neumaier derives the following approach ([1], p18, section 10):

Theorem 8.3. *Let θ^* be any value of the parameter satisfying $EX = v(\theta^*)$, $\psi : I \to \mathbb{R}$ a strictly concave, differentiable function defined on an interval I containing the range of all the functions $v_j(\theta)$ and set*

$$f(x, \theta) := \sum\nolimits_j \alpha_j[\psi(v_j(\theta) + \psi'(v_j(\theta)(x_j - v_j(\theta)]$$

with positive weights α_j. Then the expectation $g(\theta) := Ef(X, \theta)$ is bounded below and any global minimizer $\hat\theta$ of g satisfies

$$v_j(\hat\theta) = v_j(\theta^*), \quad \forall j = 1, \dots, n.$$

Remark. The idea here is that we will replace the unknown function $g(\theta)$ with $f(x, \theta)$, where x is the single realization of X which we have and then minimize $f(x, \theta)$ instead. Since f is linear in the variable x we have

$$g(\theta) = f(EX, \theta) = \sum\nolimits_j \alpha_j[\psi(v_j(\theta) + \psi'(v_j(\theta)(EX_j - v_j(\theta)] \tag{55}$$

and this approach has some merit if many of the realizations x_j are close to the mean EX_j.

If the values $v_j(\theta)$, $j = 1, \dots, n$, determine the parameter θ uniquely, the conclusion of the theorem can be strengthened to $\hat\theta = \theta^*$, that is, the minimization of $g(\theta)$ identifies the true parameter θ^*.

Proof. The strict concavity of ψ implies that at any point $z \in I$ the graph of ψ is below its tangent at the point z and is strictly below this tangent at all points $w \neq z$. More formally we have

$$f(w) \leq f(z) + f'(z)(w - z)), \quad z, w \in I, \tag{56}$$

with equality only in case of $w = z$. Now, for any θ we have

$$\begin{aligned} g(\theta) &= \sum\nolimits_j \alpha_j [\psi(v_j(\theta) + \psi'(v_j(\theta)(EX_j - v_j(\theta)] \\ &= \sum\nolimits_j \alpha_j [\psi(v_j(\theta) + \psi'(v_j(\theta)(v_j(\theta^*) - v_j(\theta)] \end{aligned}$$

and in particular $g(\theta^*) = \sum_j \alpha_j \psi(v_j(\theta^*))$. Subtraction yields

$$g(\theta) - g(\theta^*) = \sum\nolimits_j \alpha_j [\psi(v_j(\theta) + \psi'(v_j(\theta)(v_j(\theta^*) - v_j(\theta)) - \psi(v_j(\theta^*))],$$

where each expression in square brackets is nonnegative and is strictly positive if $v_j(\theta) \neq v_j(\theta^*)$. I.e.

$$g(\theta) \geq g(\theta^*)$$

with equality if and only if $v_j(\theta) = v_j(\theta^*)$, for all $j = 1, \ldots, n$. ▮

Application to our problem. Now we apply this to our problem. Here $X = (c_1^2, \ldots, c_r^2)$, where c is the vector $c = U'b$ in the problem $Ax = b$ and $A = UDV'$ is the SVD of the matrix A. Recall that we had

$$EX_j = Ec_j^2 = \sigma^2 + \tau^2 \sigma_j^{2(p+1)}$$

and we will treat the slightly more general case

$$EX_j = Ec_j^2 = v_j(\sigma, \tau) := \sigma^2 \mu_j + \tau^2 \lambda_j \tag{57}$$

with given constants μ_j and λ_j, $j = 1, \ldots, r = rank(A)$. The special case we are interested in is the case

$$\mu_j = 1 \quad \text{and} \quad \lambda_j = \sigma_j^{2(p+1)}. \tag{58}$$

Recall that we need only the parameter

$$t := \Delta^2 = \sigma^2 / \tau^2 \tag{59}$$

which is positive and we can thus work with the function $\psi(u) = \log(u)$ in theorem 8.3. Fix weights $\alpha_j > 0$ and normalize them so that

$$\alpha := \sum\nolimits_j \alpha_j = 1. \tag{60}$$

We now apply theorem 8.3 with

$$v_j(\theta) = v_j(\sigma^2, \tau^2) = \sigma^2 \mu_j + \tau^2 \lambda_j = \tau^2 [\lambda_j + t\mu_j].$$

Write $f(c, \theta)$ instead of $f(X, \theta)$. Then

$$\begin{aligned} g(\theta) = Ef(c,\theta) &= \sum\nolimits_j \alpha_j [\log(v_j(\theta)) + v_j(\theta)^{-1}(Ec_j^2 - v_j(\theta)] \\ &= \alpha + 2\alpha \log(\tau) + \sum\nolimits_j \alpha_j \log(\lambda_j + t\mu_j) + \tau^{-2} \sum\nolimits_j \alpha_j \frac{Ec_j^2}{\lambda_j + t\mu_j} \\ &:= g(\tau, t). \end{aligned}$$

Here we have reparametrized our problem in terms of the variables τ and $t = \sigma^2/\tau^2$ and need to minimize the function $g(\tau, t)$. To do this we will set the partial derivative $\partial g/\partial \tau$ equal to zero and solve for τ in terms of t yielding a path $\tau = \tau(t)$. The minimum of g is assumed along that path and we have thus reduced the problem to the minimization of a function

$$g(t) = g(t, \tau(t))$$

of one variable t only. Indeed, using $\alpha = 1$, we have

$$0 = \partial g/\partial \tau = \frac{2}{\tau} - \frac{2}{\tau^3} \sum\nolimits_j \alpha_j \frac{Ec_j^2}{\lambda_j + t\mu_j}$$

yielding

$$\tau^2 = \sum\nolimits_j \alpha_j \frac{Ec_j^2}{\lambda_j + t\mu_j}. \tag{61}$$

Putting this into the function $g(\tau, t)$ yields

$$g(t) = 2 + \sum\nolimits_j \alpha_j \log(\lambda_j + t\mu_j) + \log \left[\sum\nolimits_j \alpha_j \frac{Ec_j^2}{\lambda_j + t\mu_j} \right]. \tag{62}$$

From scaling arguments Neumaier makes the case that the weights α_j should all be equal (hence $\alpha_j = 1/r$, $j = 1, \ldots, r$), so that (neglecting constant summands) the objective function assumes the form

$$\hat{g}(t) = \log Q(t) + R(t) \quad \text{where} \tag{63}$$

$$Q(t) = \sum\nolimits_j \frac{Ec_j^2}{\lambda_j + t\mu_j} \quad \text{and} \tag{64}$$

$$R(t) = \frac{1}{r} \sum\nolimits_{j=1}^{r} \log(\lambda_j + t\mu_j). \tag{65}$$

This function is called the *generalized maximum likelihood* (GML) score function. Here $r = rank(A)$ is the number of nonzero singular values of A.

In practice the expectations Ec_j^2 are replaced with the realizations c_j^2 and the function $g(t)$ above minimized numerically. We will do this by setting the derivative $g'(t) = 0$ and solve this equation using Newton's method. As a suitable starting point for the minimization Neumaier gives

$$t := median(\lambda_j/\mu_j)$$

which is $t = median(\sigma_j^{2(p+1)})$ in our case. To solve $\hat{g}'(t) = 0$ with Newtons method we need both derivatives $\hat{g}'(t)$ and $\hat{g}''(t)$. Note first that

$$\hat{g}'(t) = \frac{Q'(t)}{Q(t)} + R'(t) \quad \text{where} \tag{66}$$

$$Q'(t) = -\sum\nolimits_j \frac{\mu_j E c_j^2}{(\lambda_j + \mu_j t)^2} \quad \text{and} \tag{67}$$

$$R'(t) = \frac{1}{r} \sum\nolimits_j \frac{\mu_j}{\lambda_j + t\mu_j}. \tag{68}$$

From this it follows that

$$\hat{g}''(t) = \frac{Q''(t)Q(t) - Q'(t)^2}{Q(t)^2} + R''(t) \quad \text{where} \tag{69}$$

$$Q''(t) = 2\sum\nolimits_j \frac{\mu_j^2 E c_j^2}{(\lambda_j + \mu_j t)^3} \quad \text{and} \tag{70}$$

$$R''(t) = -\frac{1}{r} \sum\nolimits_j \frac{\mu_j^2}{(\lambda_j + t\mu_j)^2}. \tag{71}$$

It is also recommended that the smoothness parameter p should be estimated by minimizing $\hat{g}(t)$ as a function of p also, although that is not supported by developments in the paper.

Concluding remarks. The motivation of the choice of smoothing matrix S from the problem of numeric differentiation is reasonable. This leads to the suppression of the singular values in the standard solution (50) by means of powers $\sigma_j^{2(p+1)}$.

On the other hand the stochastic heuristic for choosing the regularization parameter Δ is rather weak and nonexistent for choosing the smoothing parameter p. Here we could simply try solutions for various values of these parameters and check which ones produce the best results in terms of $\|Ax - b\|$. These are $O(n^2)$ operations whereas the SVD is much more expensive than that.

In that case the above development could be used to locate an interval for t in which a grid search for the optimal solution of $Ax = b$ is conducted.

9 Kullback-Leibler distance

Consider the uniform discrete probability distribution $p = (p_j)_{1 \leq j \leq n}$ on the set $\Omega = \{\, \omega_1, \omega_2, \ldots, \omega_n \,\}$:

$$p_j = P(\{\omega_j\}) = 1/n.$$

If $x = (x_j)$ with $x_j > 0$ and $\sum x_j = 1$ is another probability distribution on Ω, then the *Kullback-Leibler* distance $d_{KL}(x, p)$ of x from p is defined as

$$d_{KL}(x, p) = \sum\nolimits_j p_j \log(p_j/x_j) = -log(n) - \frac{1}{n} \sum\nolimits_j \log(x_j). \tag{72}$$

This function is convex in the variable x and also symmetric in x. The symmetry uses the fact that the p_j are all equal and will be used for the analytic solution of the minimization problems below. Note that we have

$$\nabla d_{KL}(x,p) = -\tfrac{1}{n}(1/x_1, 1/x_2, \ldots, 1/x_n)' \quad \text{and} \tag{73}$$

$$\nabla^2 d_{KL}(x,p) = \tfrac{1}{n} diag(1/x_1^2, 1/x_2^2, \ldots, 1/x_n^2), \tag{74}$$

where $diag(\lambda)$ denotes the diagonal matrix with the vector λ on the diagonal as usual.

9.1 Minimization of d_{KL} under probability constraints

Now let $A_k \subseteq \Omega$, $k = 1, \ldots, m$ be *disjoint* events (subsets) and consider the convex minimization problem

$$x^* = argmin\{\, d_{KL}(x,p) \;:\; P^x(A_k) = q_k \,\}. \tag{75}$$

Here $P^x(A) = E^x[1_A]$ denotes the probability of the event A under the discrete probability distribution $x = (x_j)$ on the set Ω. Note that a constraint on the probabilities x of the form $P^x(A) = r$ has the form

$$r = P^x(A) = \sum\nolimits_j x_j 1_A(\omega_j)$$

and is therefore a linear constraint in the variable x. Moreover the right hand side is a symmetric function of the variables x_j. Consequently the solution x^* of (75) must be symmetric under all permutations of coordinates which leave the sets A_k invariant, in other words the probability function

$$x^* : \omega_j \mapsto x_j^* = P^*(\omega_j)$$

is constant on all the sets A_k as well as the complement $D = [\cup A_k]^c$. This uses the fact that the A_k are disjoint since this implies that points $\omega \in \Omega$ which are in the same set A_k or are in D cannot be distinguished by the conditions $\omega \in A_k$ (i.e. if it is only determined in which of the sets A_k they are).

More formally the system of constraints

$$r_k = P^x(A_k) = \sum\nolimits_j x_j 1_{A_k}(\omega_j) \tag{76}$$

is invariant under all permutations of the variables x_j which (when applied to the points ω_j) leave the sets A_k invariant. Thus the solution x^* has the form

$$x_j^* = \begin{cases} q_k & \text{if } j \in A_k \\ q_* & \text{if } j \in D \end{cases}$$

and the variables q_k, q_* can be computed from the following system of equations

$$r_k = P^{x^*}(A_k) = q_k|A_k|$$
$$1 - \sum\nolimits_k r_k = P^{x^*}(D) = q_*|D|$$

or explicitly

$$x_j^* = \begin{cases} r_k/|A_k| & \text{if } \omega_j \in A_k \\ \frac{1}{|D|}\left(1-\sum_k r_k\right) & \text{if } \omega_j \in D. \end{cases} \tag{77}$$

Here $|D|$ denotes the cardinality of the set D as usual.

10 Pitfalls

10.1 Tolerance in the BarrierSolver too small

Recall that the BarrierSolver increases the parameter t along the central path until the upper bound on the duality gap

$$\text{dualityGap } \leq \#(\text{InequalityConstraints})/t < \mathit{tol}$$

is below the tolerance *tol* which we set. The significance of this tolerance lies in the following inequality

$$f(x) < \min(f) + \mathit{tol}, \tag{78}$$

where f is the objective function and $x = x(t)$ the point on the central path corresponding to the parameter t. Now as this parameter t increases the KKT system becomes increasingly ill conditioned up to a point where it is classified as singular and can no longer be solved.

The solution to this problem is to increase the tolerance and live with a larger duality gap and weaker estimate (78).

References

[1] Arnold Neumaier, *Solving ill conditioned and singular systems, a tutorial on regularization*, `https://www.mat.univie.ac.at/~neum/regul.html`, Department of Mathematics, University of Vienna.