Supplement for Algorithms for Estimating Trends in Global Temperature Volatility

Author 1 and Author 2
Address line
Address line

A PDIP for ℓ_1 Trend Filtering of variance

In this appendix we provide more details on how to solve the optimization problem with the objective specified in equation 2 using PDIP. The objective function is convex but not differentiable. Therefore, to be able to use PDIP we first need to derive the dual of this problem. We note that this is a generalized LASSO problem (Tibshirani and Taylor 2011). The dual of a generalized LASSO with the objective $f(x) + \lambda \|Dx\|_1$ is:

$$\min_{\nu} \quad f^*(-D^\top \nu) \qquad \text{s.t.} \quad \|\nu\|_\infty \leq \lambda$$

where $f^*(\cdot)$ is the Fenchel conjugate of f: $f^*(u) = \max_x u^\top x - f(x)$. It is simple to show that for the objective function of Equation 2

$$f^*(u) = \sum_t (u_t - 1) \log \frac{y_t^2}{1-u_t} + u_t - 1.$$

Each iteration of PDIP involves computing a search direction by taking a Newton step for the system of nonlinear equations $r_w(v, \mu_1, \mu_2) = 0$, where $w > 0$ is a parameter and

$$\begin{aligned} & r_w(v, \mu_1, \mu_2) \\ & := \begin{bmatrix} r_{dual} \\ r_{cent} \end{bmatrix} \\ & = \begin{bmatrix} \nabla f^*(-D^\top v) + \mu_1 - \mu_2 \\ -\mu_1(v - \lambda \mathbf{1}) + \mu_2(v + \lambda \mathbf{1}) - w^{-1}\mathbf{1} \end{bmatrix} \end{aligned}$$

where μ_1 and μ_2 are dual variables for the ℓ_∞ constraint. Let $A = [\nabla r_{dual}^\top, \nabla r_{cent}^\top]^\top$. The newton step takes the following form

$$r_w(v, \mu_1, \mu_2) + A \begin{bmatrix} \nabla v \\ \nabla \mu_1 \\ \nabla \mu_2 \end{bmatrix} = 0$$

with

$$A = \begin{bmatrix} \nabla^2 f^*(-D^\top v) & I & -I \\ -\mathbf{diag}(\mu_1)\mathbf{1} & -v + \lambda \mathbf{1} & \mathbf{0} \\ \mathbf{diag}(\mu_2)\mathbf{1} & v + \lambda \mathbf{1} & \mathbf{0} \end{bmatrix}.$$

Therefore, to perform the Newton step we need to compute $\nabla f^*(-D^\top v)$ and $\nabla^2 f^*(-D^\top v)$.

It is straightforward to show that

$$\begin{aligned} \nabla f^*(-D^\top v) &= -\nabla_u f^*(u) D^\top, \\ u &= -D^\top v, \\ (\nabla_u f^*(u))_j &= \log\left(\frac{y_j^2}{1-u_j}\right), \\ \nabla^2 f^*(-D^\top v) &= D \nabla_u^2 f^*(u) D^\top, \\ (\nabla_u^2 f^*(u))_j &= \mathbf{diag}\left(\frac{1}{1-u_j}\right). \end{aligned}$$

Having computed the conjugate function and its gradient and Jacobian, now we can use a number of convex optimization software packages which have an implementation of PDIP to solve the optimization problem with the objective function (2). We chose the python API of the `cvxopt` package (Andersen, Dahl, and Vandenberghe 2013).

B PDIP Update in Algorithm 1

In this section we give more details on performing the x-update step in Algorithm 1. We need to solve the following optimization problem:

$$\hat{x} := \operatorname*{argmin}_x \left(\sum_{j=1}^{n_b} (x_j + y_j^2 e^{-x_j}) + (\rho/2)\|x - \tilde{z} + u\|_2^2 + \Lambda^\top |Dx| \right)$$

where n_b is the number of local variables in each sub-cube in Figure 1, and for ease of notation we have dropped the subscript i and superscript m.

The matrix D has the following form: $D = [D_{temp} | D_{spat}]$. The matrix D_{temp} is the following block-diagonal matrix and corresponds to the temporal penalty:

$$D_{temp} = \begin{bmatrix} D_t & & \\ & \ddots & \\ & & D_t \end{bmatrix},$$

where D_t was first introduced in Section 2 of the main text and has the following form:

$$D_t = \begin{bmatrix} 1 & -2 & 1 & & & & \\ & 1 & -2 & 1 & & & \\ & & \ddots & \ddots & \ddots & & \\ & & & 1 & -2 & 1 & \\ & & & & 1 & -2 & 1 \end{bmatrix}.$$

The number of the diagonal blocks in D_{temp} is equal to the grid size $n_r \times n_c$. Each row of the matrix D_{spat} corresponds to one spatial constraint in Equation (3) in the text. For example, the first T rows correspond to $|h_{11t} - h_{21t}|$ for $t = 1, ..., T$, the next T rows correspond to $|h_{11t} - h_{12t}|$, and so on.

This optimization problem, is again a generalized LASSO problem with $f(x) = \sum_{j=1}^{n_b}(x_j + y_j^2 e^{-x_j}) + (\rho/2)\|x - \tilde{z} + u\|_2^2$.

As it was explained in Appendix A, the dual of this optimization problem is: $\min_\nu f^*(-D^\top \nu)$ with the constraints $|\nu_k| \leq \Lambda_k$. To use PDIP we first need to compute the conjugate function $f^*(\cdot)$. We have:

$$\begin{aligned} & f^*(\xi) \\ & = \max_x \xi^\top x - f(x) \\ & = \max_x \sum_{j=1}^{n_b}(\xi_j x_j - x_j - y_j^2 e^{-x_j} - (\rho/2)(x_j - \tilde{z}_j + u_j)). \end{aligned}$$

Setting the derivative of the terms inside the summation to 0, we obtain:

$$\xi_j - y_j^2 e^{-x_j^*} - \rho x_j^* + \rho(\tilde{z}_j - u_j) = 0, \qquad (1)$$

where x^* is the maximizer in B. Then, it can be shown that x_j^* which satisfies (1) can be obtained as follows:

$$\begin{aligned} x_j^* &= \mathscr{W}\left(\frac{y_j^2}{\rho} e^{\phi_j}\right) - \phi_j, \\ \phi_j &= \frac{1 - \xi_j - \rho(\tilde{z}_j - u_j)}{\rho}. \end{aligned}$$

In this equation, $\mathscr{W}(\cdot)$ is the Lambert W function (Corless et al. 1996). Finally, the conjugate function is: $f^*(\xi) = \sum_{j=1}^{n_b}(\xi_j x_j^* - x_j^* - y_j^2 e^{-x_j^*} - (\rho/2)(x_j^* - \tilde{z}_j + u_j))$.

To use PDIP, we also need to evaluate ∇f^* and $\nabla^2 f^*$. First note that $\frac{\partial \mathscr{W}(q)}{\partial q} = \frac{\mathscr{W}(q)}{q(1+\mathscr{W}(q))}$ and $\frac{\partial^2 \mathscr{W}(q)}{\partial q^2} = -\frac{\mathscr{W}^2(q)(\mathscr{W}(q)+q)}{q^2(1+\mathscr{W}(q))^3}$. Using the chain rule we get:

$$\frac{\partial f^*(\xi)}{\partial \xi_j} = x_j^* + \frac{\partial x_j^*}{\partial \xi_j}\left[\xi_j - 1 + y_j^2 e^{-x_j^*} + \rho(\tilde{z}_j - u_j - x_j^*)\right],$$

where we have

$$\frac{\partial x_j^*}{\partial \xi_j} = \frac{1}{\rho(1 + \mathscr{W}((y_j^2/\rho)e^{-\phi_j}))}.$$

By some tedious but straightforward computation we can obtain the second derivatives:

$$\begin{aligned} \frac{\partial^2 f^*(\xi)}{\partial \xi_j^2} &= \frac{\partial x_j^*}{\partial \xi_j} - \rho \frac{\partial^2 x_j^*}{\partial \xi_j^2}\left[\phi_j + x_j^* - \tilde{z}_j + u_j\right], \\ &+ \frac{\partial x_j^*}{\partial \xi_j}\left[1 - y_j^2 \frac{\partial x_j^*}{\partial \xi_j} e^{-x_j^*} - \rho \frac{\partial x_j^*}{\partial \xi_j}\right], \\ \frac{\partial^2 x_j^*}{\partial \xi_j^2} &= \frac{\mathscr{W}((y_j^2/\rho)e^{-\phi_j})}{\rho^2(1 + \mathscr{W}((y_j^2/\rho)e^{-\phi_j}))^3}. \end{aligned}$$

C Proof of Lemma 1

Proof. If $f(x) = \sum_k f_k(x_k)$ then $[\mathbf{prox}_{\mu f}(x)]_k = \mathbf{prox}_{\mu f_k}(u_k)$. So $[\mathbf{prox}_{\mu f}(u)]_k = \min_{x_k} \mu\big(x_k + y_k^2 e^{-x_k}\big) + \frac{1}{2}(x_k - u_k)^2$. Setting the derivative to 0 and solving for u_k gives the result. Similarly, $[\mathbf{prox}_{\rho g}(u)]_\ell = \rho\lambda_\ell|z_\ell| + 1/2(z_\ell - u_\ell)^2$. This is not differentiable, but the solution must satisfy $\rho \cdot \lambda_\ell \cdot \partial\big(|z_\ell|\big) = u_\ell - z_\ell$ where $\partial\big(|z_\ell|\big)$ is the sub-differential of $|z_\ell|$. The solution is the soft-thresholding operator $S_{\rho\lambda_\ell}(u_\ell)$. $\square$

References

Andersen, M. S.; Dahl, J.; and Vandenberghe, L. 2013. *CVXOPT: A Python package for convex optimization, version 1.1.6.* Available at `cvxopt.org`.

Corless, R. M.; Gonnet, G. H.; Hare, D. E. G.; Jeffrey, D. J.; and Knuth, D. E. 1996. On the LambertW function. *Advances in Computational Mathematics* 5(1):329–359.

Tibshirani, R. J., and Taylor, J. 2011. The solution path of the generalized lasso. *Annals of Statistics* 39(3):1335–1371.