WPS-044 18/09/2010

A note on 'good starting values' in numerical optimisation

M. Gilli
E. Schumann

A note on ‘good starting values’ in numerical optimisation

Manfred Gilli and Enrico Schumann[*]

3 June 2010

Abstract

Many optimisation problems in finance and economics have multiple local optima or discontinuities in their objective functions. In such cases it is stressed that ‘good starting points are important’. We look into a particular example: calibrating a yield curve model. We find that while ‘good starting values’ suggested in the literature produce parameters that are indeed ‘good’, a simple best-of-N–restarts strategy with random starting points gives results that are never worse, but better in many cases.

Many optimisation problems in finance and economics are difficult to solve. These models have multiple local optima or discontinuities in their objective functions. In such cases it is often stressed that ‘good starting points are important’. This statement in itself is both trivial and useless: trivial if it is to imply that there exist good starting values (just pick the global minimum as the starting value); useless if no further advice is given how to find such good starting values.

Starting values can often be derived from economic or mathematical intuition about the problem. But the subtleties of numerical representation and the iterative nature of optimisation algorithms make it unpredictable whether a ‘good starting point’ really leads to the global optimum or not (see Nash, 1990, pp. 146–147; McCullough and Vinod, 1999; McCullough and Renfro, 2000; McCullough and Vinod, 2003 and the various responses; McKinnon, 1998). Even a run from a starting point close to the global optimum is practically not guaranteed to converge.

But fortunately, we do not need to know good starting points. We can simply restart the algorithm from different starting values, and then keep the best result. In this note, we demonstrate such an experiment, calibrating a yield structure model. We show that while ‘good starting values’ suggested in the literature produce parameters that are indeed ‘good’, a simple best-of-N–restarts strategy with random starting points gives results that are never worse, but better in many cases.

[*]University of Geneva, Switzerland. Corresponding author: es AT vipag.com. We gratefully acknowledge financial support from the EU Commission through MRTN-CT-2006-034270 COMISEF.

An experiment

The Nelson–Siegel–Svensson model is widely used by central banks and other market participants as a model of the term structure of interest rates. Let $y(\tau)$ be the zero rate for maturity τ, then the model defines such a rate as

$$\begin{aligned} y(\tau) = \beta_1 + \beta_2 \left[\frac{1-\exp(-\tau/\lambda_1)}{\tau/\lambda_1} \right] + \\ \beta_3 \left[\frac{1-\exp(-\tau/\lambda_1)}{\tau/\lambda_1} - \exp(-\tau/\lambda_1) \right] + \beta_4 \left[\frac{1-\exp(-\tau/\lambda_2)}{\tau/\lambda_2} - \exp(-\tau/\lambda_2) \right] . \end{aligned} \tag{1}$$

We will not discuss the details of the model here; see Nelson and Siegel (1987), Svensson (1994), or Gilli et al. (2010). We need to estimate six parameters: β_1, β_2, β_3, β_4, λ_1 and λ_2. Estimates can be obtained by minimising the difference between the model rates y, and observed rates y^M where the superscript stands for 'market' (such y^M can for instance be computed by bootstrapping). We use the data from Diebold and Li (2006), obtained from `http://www.ssc.upenn.edu/~fdiebold/papers/paper49/FBFITTED.txt`. The data set consists of monthly zero rates for U.S. bonds for 18 different maturities: 1/12, 3/12, 6/12, 9/12, 1, 1 1/4, 1 1/2, 1 3/4, 2, 2 1/2, 3, ..., 10 years, so we have $y^M(1/12)$, $y^M(3/12)$, and so on; see Diebold and Li (2006) for a detailed description. Altogether, there are 372 cross-sections of yields, from January 1970 to December 2000.

An optimisation problem can then be stated as

$$\min_{\beta,\,\lambda} \sum \left(y - y^M\right)^2 . \tag{2}$$

We constrain the parameters to the ranges

$$\begin{aligned} &0 \leq \beta_1 \leq 15, \quad -15 \leq \beta_2 \leq 30, \quad -30 \leq \beta_3 \leq 30, \quad -30 \leq \beta_4 \leq 30, \\ &0 \leq \lambda_1 \leq 3, \quad 3 \leq \lambda_2 \leq 6 , \end{aligned}$$

see Gilli et al. (2010) for a discussion.

This optimisation problem typically has many local minima, so classical techniques based on derivatives of the objective function are not appropriate. But still, researchers and operators use such techniques. There exist various prescriptions for how to choose the starting value of an optimisation, see Gimeno and Nave (2009) or Manousopoulos and Michalopoulos (2009). Here, as described in Manousopoulos and Michalopoulos (2009, p. 598), we use starting points as follows:

$$\begin{aligned}
\beta_1 &= (y^M(9) + y^M(10))/2 \\
\beta_2 &= y^M(1/12) - \beta_1 \\
\beta_3 &= 0 \\
\beta_4 &= 0 \\
\lambda_1 &= 1 \\
\lambda_2 &= 1
\end{aligned}$$

We tested other variants, but the results were essentially unchanged; R-code to replicate the experiment is given in the appendix.

To demonstrate our point, we use the function `nlminb` from R's `stats` package (R Development Core Team, 2008) and fit model (2): so for each month, we have 18 observed zero rates, and we wish to obtain parameters such that Equation (1) closely approximates these rates. We try two optimisation strategies: firstly, we run an optimisation with 'good starting values' (the GSV strategy). We also run 100 optimisations with starting values that are randomly chosen from the feasible ranges, and then pick the best of these solutions; we call this the best-of-N (BON), strategy, here best-of-100. This may not seem a fair comparison: the computing time for BON will on average be 100 times greater than for GSV. Yet fast computation is never a goal in itself; it is a practical constraint. In other words, allowed computing time determines N. And here, 100 runs are not too expensive: they take less than 5 seconds in R 2.10.1 on an Intel P8700 (single core) at 2.53 GHz with 2 GB RAM.

For each month, we so obtain a solution for GSV, and a solution for BON. Then, we compute the root mean squared error (rms) of every solution as

$$\sqrt{\frac{1}{m}\sum_{i=1}^{m}\Big(y^M(\tau_i) - y(\tau_i)\Big)^2}\,. \tag{3}$$

This equation is equivalent to our objective function, but it rescales the objective function to make the numerical results interpretable. In our data set, m is 18; we give all results in basis points (bp). Next, we compute the difference between the rms of the best solution of the 100 runs with random starting points, and the rms obtained with 'good starting values', ie,

$$\text{rms}_{\text{BON}} - \text{rms}_{\text{GSV}}\,.$$

These differences are plotted in Figure 1; each dot shows the error difference for one month. If this difference is positive, the GSV run gave a better result than the BON run; if the difference is negative, the BON run was better. The maximum

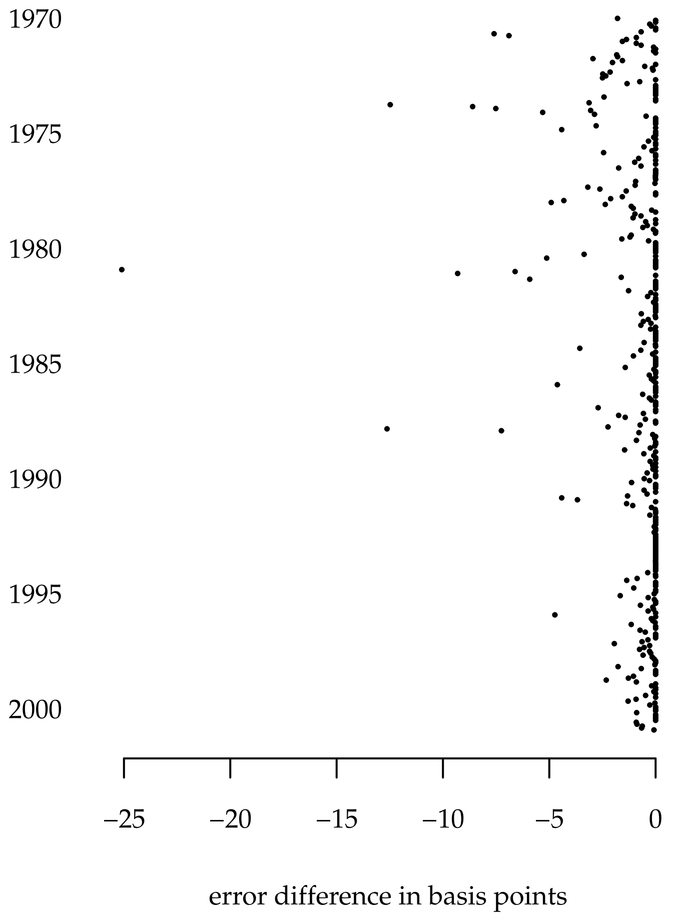

Figure 1: Error difference in basis points between best-of-100–restarts (BON) solutions and 'good starting value' (GSV) solutions. A negative difference means the best-of-100 strategy worked better; a positive difference (there is none) means the GSV strategy worked better.

difference is 0.0 bp, thus the BON solutions were never worse than the GSV solutions. The minimum difference is 25.1 bp, so here the BON run yielded a solution with an error that was substantially smaller. Altogether, the improvements seem small in most cases, and we certainly need to judge their relevance from the view of a concrete application; but for instance the bid–ask spread for liquid government bonds like those of Germany is, in terms of yield, often only a fraction of one basis point.

Conclusion

In our example, 'good starting values' have lead to good solutions indeed, but the best-of-100–restarts strategy with random starting points gave results that were never worse, but better in many cases.

The important point is that trying different starting points does not cost us much. True, we need more computing time. Yet the fast solution from a single restart can only be obtained by trading off solution quality against speed. We need to judge with respect to our specific application how much computing time we can afford.

In any case, rerunning an optimisation with different starting values is a robustness check that should always be applied to optimisation routines. If different starting values lead to different solutions, then a multiple-restart strategy should always be preferred (we can always include the 'good starting point' in our set of starting values). Or better yet, we take it as a sign that alternative methods like heuristics (Gilli and Winker, 2009) are more appropriate, anyway.

References

Francis X. Diebold and Canlin Li. Forecasting the Term Structure of Government Bond Yields. *Journal of Econometrics*, 130(2):337–364, 2006.

Manfred Gilli and Peter Winker. Heuristic optimization methods in econometrics. In David A. Belsley and Erricos Kontoghiorghes, editors, *Handbook of Computational Econometrics*. Wiley, 2009.

Manfred Gilli, Stefan Große, and Enrico Schumann. Calibrating the Nelson–Siegel–Svensson Model. *COMISEF Working Paper Series No. 31*, 2010. available from `http://comisef.eu/?q=working_papers`.

Ricardo Gimeno and Juan M. Nave. A Genetic Algorithm Estimation of the Term Structure of Interest Rates. *Computational Statistics & Data Analysis*, 53:2236–2250, 2009.

Polychronis Manousopoulos and Michalis Michalopoulos. Comparison of Non-linear Optimization Algorithms for Yield Curve Estimation. *European Journal of Operational Research*, 192:594–602, 2009.

B.D. McCullough and Charles G. Renfro. Some Numerical Aspects of Nonlinear Estimation. *Journal of Economic and Social Measurement*, 26(1):63–77, 2000.

B.D. McCullough and H.D. Vinod. The Numerical Reliability of Econometric Software. *Journal of Economic Literature*, 37(2):633–665, June 1999.

B.D. McCullough and H.D. Vinod. Verifying the Solution from a Nonlinear Solver: A Case Study. *American Economic Review*, 93(3):873–892, 2003.

K.I.M. McKinnon. Convergence of the Nelder–Mead Simplex Method to a Nonstationary Point. *SIAM Journal on Optimization*, 9(1):148–158, 1998.

John C. Nash. *Compact Numerical Methods for Computers: Linear Algebra and Function Minimization.* Adam Hilger, 2nd edition, 1990.

Charles R. Nelson and Andrew F. Siegel. Parsimonious Modeling of Yield Curves. *Journal of Business*, 60(4):473–489, 1987.

R Development Core Team. *R: A Language and Environment for Statistical Computing.* R Foundation for Statistical Computing, Vienna, Austria, 2008. URL `http://www.R-project.org`. ISBN 3-900051-07-0.

Lars E.O. Svensson. Estimating and Interpreting Forward Interest Rates: Sweden 1992–1994. *IMF Working Paper 94/114*, 1994.

A R code

The complete experiment, including the download of the data set from Diebold and Li (2006), can be replicated with the following code.

```r
# define Nelson--Siegel--Svensson model
NSS <- function(betaV,mats) {
    # betaV = beta1-4, lambda1-2
    gam1 <- mats / betaV[5]
    gam2 <- mats / betaV[6]
    aux1 <- 1 - exp(-gam1)
    aux2 <- 1 - exp(-gam2)
    y <- betaV[1] + betaV[2] * (aux1 / gam1) +
            betaV[3] * (aux1 / gam1 + aux1 - 1) +
            betaV[4] * (aux2 / gam2 + aux2 - 1)
    return(y)
}

# define objective function
OF <- function(betaV,dataList) {
    mats  <- dataList$mats
    yM    <- dataList$yM
    model <- dataList$model
    y     <- model(betaV,mats)
    aux   <- y - yM
    aux   <- crossprod(aux)
    return(aux)
}

# get bliss/diebold/li data
x <- url("http://www.ssc.upenn.edu/~fdiebold/papers/paper49/FBFITTED.txt")
open(x); dili <- scan(x, skip = 14); close(x)
mat <- NULL
for (i in 1:372) {mat <- rbind(mat,dili[(19*(i-1)+1):(19*(i-1)+19)])}
mats  <- c(1,3,6,9,12,15,18,21,24,30,36,48,60,72,84,96,108,120)/12

# the obligatory perspective plot
persp(x = mat[,1], y = mats,mat[,-1],
        phi = 40, theta = 20, ticktype = "detailed",
        xlab = "time", ylab = "time to maturity in years",
        zlab = "zero rates in %")

# settings: minimum, maximum, number of parameters
settings <- list(
        min = c( 0,-15,-30,-30,0,3),
        max = c(15, 30, 30, 30,3,6),
        d = 6)

# how many restarts per month?
trials <- 100

# set array to store results
res <- array(NA, c(372,trials));goodRes <- array(NA, c(372,1))

# run through all months
set.seed(75325428)
howFar <- txtProgressBar(min=1,max=372,style=3)
for(t in 1:372){
    # market yields
    yM <- as.numeric(mat[t,-1])
    dataList <- list(yM = yM, mats = mats, model = NSS)
    # random starting values
    for( rr in seq(trials) ) {
        s0 <- settings$min +
                (settings$max - settings$min) * runif(settings$d)
        sol <- nlminb(s0, OF,
                data = dataList,
                lower = settings$min, upper = settings$max)
        res[t,rr] <- sqrt(sum((NSS(sol$par,mats)-yM)^2)/18)
    }
    # good starting values
    s0 <- c((yM[18]+yM[17])/2,yM[1]-(yM[18]+yM[17])/2,0,0,1,1)
    sol <- nlminb(s0,OF,
            data = dataList,
            lower = settings$min, upper = settings$max,
            control = list(eval.max=50000,iter.max=50000) )
    goodRes[t] <- sqrt(sum((NSS(sol$par,mats)-yM)^2)/18)
    #
    setTxtProgressBar(howFar, value=t)
}
close(howFar)
# compute difference in basis points between random-start solutions and good-
    start solutions
diffs <- 100 * (apply(res,1,min) - goodRes)
labs <-  rep(NA,372); labs[seq(from=1, to= 372, by = 60)] <- seq(1970,2000,5)
diffs <- diffs[372:1]; labs <- labs[372:1] # flip upside down for plotting

# plot
#setEPS()
#postscript("errors.eps",family="Palatino",width=4, height=5)
par(bty="n",las=1,ps=10,mar=c(4,4,1,1))
plot(diffs,1:372,type="n",xlab="error difference in basis points",yaxt="n",
    ylab="")
points(diffs,1:372,cex=.25,pch=19)
axis(2,at=1:372,labels=labs,lty=0)
#dev.off()
```