AARHUS
UNIVERSITY
BUSINESS AND SOCIAL SCIENCES
DEPARTMENT OF ECONOMICS AND BUSINESS

CREATES
Center for Research in Econometric Analysis of Time Series

Conditionally-Uniform Feasible Grid Search Algorithm

Matt P. Dziubinski

CREATES Research Paper 2012-03

Department of Economics and Business
Aarhus University
Bartholins Allé 10
DK-8000 Aarhus C
Denmark

Email: oekonomi@au.dk
Tel: +45 8716 5515

CONDITIONALLY-UNIFORM FEASIBLE GRID SEARCH ALGORITHM

MATT P. DZIUBINSKI

Abstract. We present and evaluate a numerical optimization method (together with an algorithm for choosing the starting values) pertinent to the constrained optimization problem arising in the estimation of the GARCH models with inequality constraints, in particular the Simplified Component GARCH Model (SCGARCH), together with algorithms for the objective function and analytical gradient computation for SCGARCH.

JEL Classification. C32, C51, C58, C61, C63, C88.

1. Introduction

In this paper we present a numerical optimization method applicable to the estimation of the Simplified Component GARCH Model (SCGARCH) of Dziubinski (2011). The method, the Conditionally-Uniform Feasible Grid Search (CUFGS), is essentially a particular kind of a random grid search coupled with a constrained feasible Sequential Quadratic Programming (SQP) algorithm.

One of the reasons for developing it are the problems encountered when using non-specialized gradient-based algorithms – due to constrained feasible space requirement and scaling. For example, in relation to the Broyden-Fletcher-Goldfarb-Shanno (BFGS) algorithm, popular with econometricians, Nocedal and Wright (2006) write:

(1) "*BFGS updating is generally less effective for constrained problems than in the unconstrained case because of the requirement of maintaining a positive definite approximation to an underlying matrix that often does not have this property.*"

Date: January 25, 2012.

2000 *Mathematics Subject Classification.* Primary 62F30, 65C60, 65Y20, 90C30, 90C55; Secondary 37M10, 62M10, 91B84.

Key words and phrases. Constrained optimization, GARCH, infeasibility, inference under constraints, nonlinear programming, performance of numerical algorithms, SCGARCH, sequential quadratic programming.

We acknowledge financial support by the Center for Research in Econometric Analysis of Time Series, CREATES, funded by the Danish National Research Foundation.

(2) "*SQP methods are most efficient if the number of active constraints is nearly as large as the number of variables, that is, if the number of free variables is relatively small. They require few evaluations of the functions, in comparison with augmented Lagrangian methods, and can be more robust on badly scaled problems.*"

We also provide the objective function and analytical gradient computation algorithms, which are useful for the practical implementation purposes.

The paper proceeds as follows. In Section 2 we present the SCGARCH model. We discuss estimation in Section 3. In Sections 4 and 5 we analyze the estimation algorithms and show the results. Section 6 contains our conclusions.

2. The Model

For reference we present the SCGARH model; for details see Dziubinski (2011). The observed time-series (e.g., log return on the spot asset price), r follows (over time steps of length $\Delta \equiv 1$) the following process under the (physical) probability measure P,

$$r_{t+1} \equiv \log \frac{S_{t+1}}{S_t} = \mu_{t+1} + \sqrt{v_{t+1}} w_{t+1} \tag{2.1}$$

$$v_{t+1} = x_{t+1} + p_v(v_t - x_t) + i_v u_{v,t} \tag{2.2}$$

$$x_{t+1} = m_x + p_x(x_t - m_x) + i_x u_{x,t} \tag{2.3}$$

with

$$\mu_{t+1} = r_f + \lambda v_{t+1} \tag{2.4}$$

$$u_{v,t} = (w_t^2 - 1) - 2g_v\sqrt{v_t} w_t \tag{2.5}$$

$$u_{x,t} = (w_t^2 - 1) \tag{2.6}$$

$$w \overset{P}{\sim} GWN(0,1) \tag{2.7}$$

where r_f is the continuously compounded interest rate for the time interval of length Δ, v_t is the conditional variance of the log return between $t-1$ and t, with $v \in \mathcal{P}$. The process w is a *Gaussian white noise* with mean 0 and variance 1, i.e., $w_t \overset{P}{\sim} \mathcal{N}(0,1) \quad \forall t \in \mathbb{T}$ and $w \overset{P}{\sim} WN(0,1)$ (under probability measure P, w has mean 0 and covariance function $\gamma(s,t) = \delta_{|t-s|}$, where $\delta_h := \mathbb{1}_{\{0\}}(h)$ is the Kronecker delta).

This model is a simplified specification of the Christoffersen et al. (2008) model, solving the problem of ensuring non-negativity of the conditional variance. The sufficient conditions for non-negativity of

volatility components v and x are:

$$p_x \leq 1, b_v > 0,\ i_v > 0,\ i_x > 0 \tag{2.8}$$

$$n_x > i_x + i_v \tag{2.9}$$

$$p_x > p_v > i_v g_v^2 > 0 \tag{2.10}$$

We denote our parameter vector by

$$\theta := (r_f,\ \lambda,\ n_x,\ i_v,\ i_x,\ p_v,\ p_x,\ g_v)^\mathsf{T}$$

and the restricted parameter space by

$$\tilde{\Theta} := \{\theta \subseteq \Theta :\ (2.8) - (2.10)\},$$

where $\Theta \subseteq \mathbb{R}^p$, $p = 8$.

3. Maximum Likelihood Estimation

A statistical method used for estimating the model is the Maximum Likelihood Estimation (MLE), which involves maximizing an objective function (called the (log)likelihood function) in order to obtain the estimates.

Dziubinski (2011) shows that we can state our optimization problem as a constrained minimization problem:

$$\hat{\theta} = \arg\min_{\theta \in \tilde{\Theta}} \tilde{Q}_N(\theta) \tag{3.1}$$

$$\tilde{Q}_N(\theta) = \sum_{t=0}^{N} l_t(\theta) \tag{3.2}$$

$$l_t(\theta) = \log(v_t) + w_t^2. \tag{3.3}$$

3.1. **Objective Function Computation.** Here we provide an algorithm to compute the objective function.[1] We can compute the summands (3.3) of (3.2) using the specification given in (2.1)–(2.7). The following procedures implement the computation given in these equations:

Summand $l_0(\theta, w, v, x, r)$
Input: a parameter vector θ, observation r_0, starting values v_0, x_0
Output: summand l_0, value w_0
1 $w_0 \leftarrow (r_0 - r_f - \lambda v_0)/v_0^{1/2}$
2 $l_0 \leftarrow \log(v_0) + w_0^2$
3 **return** l_0

[1]The conventions for pseudocode may be found on pp. 19–20 of Cormen et al. (2001). In addition to those, the evaluation strategy for modified w, v, x is *pass-by-reference* (the modifications made to these variables are preserved across calls).

SUMMAND $l_{t+1}(\theta, w, v, x, r)$
INPUT: a parameter vector θ, observation r_{t+1}, values w_t, v_t, x_t
OUTPUT: summand l_{t+1}, values w_{t+1}, v_{t+1}, x_{t+1}
1 $u_{x,t} \leftarrow (w_t^2 - 1)$
2 $u_{v,t} \leftarrow (w_t^2 - 1) - 2g_v v_t^{1/2} w_t$
3 $x_{t+1} \leftarrow n_x + p_x x_t + i_x u_{x,t}$
4 $v_{t+1} \leftarrow x_{t+1} + p_v(v_t - x_t) + i_v u_{v,t}$
5 $w_{t+1} \leftarrow (r_{t+1} - r_f - \lambda v_{t+1})/v_{t+1}^{1/2}$
6 $l_{t+1} \leftarrow \log(v_{t+1}) + w_{t+1}^2$
7 **return** l_{t+1}

Finally, the procedure OBJECTIVE-FUNCTION implements the computation of the objective function (3.2):

OBJECTIVE-FUNCTION $l(N, \theta, w, v, x, r)$
INPUT: a parameter vector θ, starting values r_0, v_0, x_0
OUTPUT: the objective function $l \equiv \sum_{t=0}^{N} l_t(\theta)$
1 $l \leftarrow$ SUMMAND $l_0(\theta, w, v, x, r)$
2 **for** $t \leftarrow 0$ **to** $(N-1)$ **do**
3 $\quad l \leftarrow l +$ SUMMAND $l_{t+1}(\theta, w, v, x, r)$
4 **return** l

3.2. **Analytical Gradient.** To implement MLE in practice it is useful to have the analytical gradient. There are at least two reasons for that. First, in case of GARCH models estimation using gradient-based optimization the analytical gradient is more accurate than its numerical approximation (see (Zivot, 2009, Section 5.1) and Brooks et al. (2001)). Second, it may also be applied for computing the outer-product gradient (OPG) estimate of the information matrix. Here is the pseudocode for the procedures implementing the computation of the analytical gradient for the objective function in our model:

SUMMAND $\nabla l_0(\theta, w, v, x, r)$
INPUT: a parameter vector θ, observation r_0, values w_0, v_0, x_0
OUTPUT: summand ∇l_0
1 $\nabla x_0 \leftarrow 0$
2 $\nabla v_0 \leftarrow 0$
3 $\nabla w_0 \leftarrow -\frac{1}{2} \cdot v_0^{-1} \cdot w_0 \cdot \nabla v_0$
4 $\partial_{r_f} w_0 \leftarrow \partial_{r_f} w_0 - v_0^{-1/2}$
5 $\partial_{\lambda} w_0 \leftarrow \partial_{\lambda} w_0 - v_0^{1/2}$
6 $\nabla w_0^2 \leftarrow 2 \cdot w_0 \cdot \nabla w_0$
7 $\nabla l_0 \leftarrow v_0^{-1} \cdot \nabla v_0 + \nabla w_0^2$
8 **return** ∇l_0

SUMMAND $\nabla l_{t+1}(\theta, w, v, x, r)$
INPUT: a parameter vector θ, observation r_{t+1}, values w_t, v_t, x_t

OUTPUT: summand ∇l_{t+1}, values w_{t+1}, v_{t+1}, x_{t+1}

$\nabla u_{x,t} \leftarrow \nabla w_t^2$
$\nabla x_{t+1} \leftarrow p_x \cdot \nabla x_t + i_x \cdot \nabla u_{x,t}$
$\partial_{n_x} x_{t+1} \leftarrow \partial_{n_x} x_{t+1} + 1$
$\partial_{i_x} x_{t+1} \leftarrow \partial_{i_x} x_{t+1} + u_{x,t}$
$\partial_{p_x} x_{t+1} \leftarrow \partial_{p_x} x_{t+1} + x_t$
$\nabla u_{v,t} \leftarrow \nabla w_t^2 + g_v \cdot w_t \cdot v_t^{-1/2} \cdot \nabla v_t + 2 \cdot g_v \cdot v_t^{1/2} \cdot \nabla w_t$
$\partial_{g_v} u_{v,t} \leftarrow \partial_{g_v} u_{v,t} + 2 \cdot v_t^{1/2} \cdot w_t$
$\nabla v_{t+1} \leftarrow \nabla x_{t+1} + p_v \cdot (\nabla v_t - \nabla x_t) + i_v \cdot \nabla u_{v,t}$
$\partial_{i_v} v_{t+1} \leftarrow \partial_{i_v} v_{t+1} + u_{v,t}$
$\partial_{p_v} v_{t+1} \leftarrow \partial_{p_v} v_{t+1} + (v_t - x_t)$
$\nabla w_{t+1} \leftarrow -\frac{1}{2} \cdot v_{t+1}^{-1} \cdot w_{t+1} \cdot \nabla v_{t+1}$
$\partial_{r_f} w_{t+1} \leftarrow \partial_{r_f} w_{t+1} - v_{t+1}^{-1/2}$
$\partial_{\lambda} w_{t+1} \leftarrow \partial_{\lambda} w_{t+1} - v_{t+1}^{1/2}$
$\nabla w_{t+1}^2 \leftarrow 2 \cdot w_{t+1} \cdot \nabla w_{t+1}$
$\nabla l_{t+1} \leftarrow v_{t+1}^{-1} \cdot \nabla v_0 + \nabla w_{t+1}^2$
return ∇l_{t+1}

Finally:

GRADIENT $\nabla l(N, \theta, w, v, x, r)$
INPUT: a parameter vector θ, starting values r_0, v_0, x_0
OUTPUT: the gradient $g \equiv \nabla l(\theta)$

$l \leftarrow$ SUMMAND $l_0(\theta, w, v, x, r)$
g $\leftarrow$ SUMMAND $\nabla l_0(\theta, w, v, x, r)$
for $t \leftarrow 0$ **to** $(N-1)$ **do**
    $l \leftarrow l +$ SUMMAND $l_{t+1}(\theta, w, v, x, r)$
    g $\leftarrow$ g + SUMMAND $\nabla l_{t+1}(\theta, w, v, x, r)$
return g

Note, that the calculation of the gradient still requires the computation of w, v and x – this is done by SUMMAND $l_t(\theta, w, v, x, r)$, $t \in \mathbb{T}$.

4. A Simulation Study of Estimation Method Choice

4.1. **Overview.** In this section we provide an overview of some of the methods to estimate our model. We simulate the model using the coefficient values given in Table 1 (they are interesting in practice, since they have the same magnitude as those in Table 1 of Dziubinski (2011)) and estimate the parameters using the simulated data and the following algorithms[2]:

[2]We use the implementations thereof provided by O2scl – an object-oriented library for numerical programming in C++ – see: `http://o2scl.sourceforge.net/`. The exception is FSQP, which uses CFSQP implementation – see Lawrence et al. (1997).

(1) NM - Nelder-Mead
(2) FR - Fletcher-Reeves
(3) PR - Polak-Ribière
(4) BFGS - Broyden-Fletcher-Goldfarb-Shanno
(5) SA - Simulated Annealing
(6) PG - Projected Gradient
(7) SPG - Spectral Projected Gradient
(8) FSQP - Feasible Sequential Quadratic Programming

N = 1,000	Simulation	Estimation Starting Values
r_f	1.000×10^{-1}	9.000×10^{-2}
λ	$2.000 \times 10^{+0}$	$1.800 \times 10^{+0}$
n_x	8.000×10^{-6}	7.200×10^{-6}
i_v	1.000×10^{-6}	9.000×10^{-7}
i_x	2.000×10^{-6}	1.800×10^{-6}
p_v	6.000×10^{-1}	5.400×10^{-1}
p_x	9.000×10^{-1}	8.100×10^{-1}
g_v	$4.000 \times 10^{+2}$	$3.600 \times 10^{+2}$
$\tilde{Q}_n$	$-8.427097 \times 10^{+3}$	$-6.228086 \times 10^{+3}$

TABLE 1. The coefficient values used in the simulation study.

The maximum number of iterations is 1,000 in all of the cases[3] The sample size $N = 1,000$.

The Nelder-Mead (also known as downhill simplex) algorithm is a derivative-free optimization method, FR and PR are nonlinear conjugate gradient methods, BFGS is a quasi-Newton method. PG is an extension of the steepest descent method for unconstrained minimization, where a line search is performed over the direction of a projected gradient. SPG is similar to PG, except accelerated convergence due to the choice of the spectral step-length; see Birgin et al. (2000) for details. SA is a probabilistic metaheuristic (convergence to an optimal solution is not guaranteed) for the global derivative-free optimization. FSQP is a quadratic programming method applicable to the constrained optimization problems. For the first four algorithms, see Nocedal and Wright (2006), for SA see Henderson et al. (2003) or Dreo et al. (2005), for PG see Kelley (1999) and for SPG see Birgin et al. (2000).

FSQP is based on Sequential Quadratic Programming (SQP), modified so as to generate feasible iterates. Sequential quadratic programming

[3]The reason for choosing this criterion as opposed to, say, maximum elapsed time, is to ensure the reproducibility of the results, independent of the performance of the computer hardware.

(SQP) methods model the general constrained optimization problem by a quadratic programming subproblem at each iterate and define the search direction to be the solution of this subproblem. It is important to design the quadratic subproblem so that it yields a good step for the nonlinear optimization problem, see Nocedal and Wright (2006). A description of FSQP can be found in Lawrence et al. (1997). There are two FSQP algorithms: FSQP-AL and FSQP-NL. In the FSQP-AL (monotone line search), an Armijo type arc search is used with the property that the step of unit length is eventually accepted, which is a requirement for superlinear convergence. In the FSQP-NL algorithm the same effect is achieved by means of a nonmonotone search along a straight line. In other words, in the FSQP-AL algorithm the objective function decreases at each iteration, while in FSQP-NL we have a decrease of the objective function within at most four iterations. For details, see Lawrence et al. (1997).

For reproducibility purposes, we use the default seed for the Simulated Annealing. To minimize the warm-up period we use the unconditional mean of v and x to initialize v_0 and x_0, that is, we set the starting values to m_x derived from the starting values of n_x and p_x.

Note, that McCullough and Renfro (1999) and Brooks et al. (2001) stress the importance of reporting the initial values provided to the GARCH estimation software. Without them, any conditional heteroscedasticity model is only partially specified, because the elements on which the likelihood is conditioned are not specified if the initial values are not given. McCullough and Renfro (1999) also report that the initialization of the series, though often overlooked, can substantially affect the "solution" produced by the software.

4.2. **Constraints.** In practice we may actually need a constrained minimization algorithm in order to ensure that the non-negativity conditions hold. Only the last three algorithms in our list are of this type (note, that PG and SPG only allow for hypercubic constraints, while FSQP allows for nonlinear functional constraints).

A solution commonly used in practice, see for example in Rouah and Vainberg (2007), is to implement a penalty, so that violating non-negativity conditions generates a large value of the objective function. For the methods requiring this modification, we adjust the algorithms as follows:

PENALIZED-OBJECTIVE-FUNCTION $l(N, \theta, w, v, x, r)$
INPUT: a parameter vector θ, starting values r_0, v_0, x_0
OUTPUT: the objective function $l \equiv \sum_{t=0}^{N} l_t(\theta)$
1 **if** POSITIVITY-CONDITIONS(θ) $\neq$ TRUE
2 **then return** PENALTY

3 $l \leftarrow$ SUMMAND $l_0(\theta, w, v, x, r)$
4 **for** $t \leftarrow 0$ **to** $(N-1)$ **do**
5 $\quad l \leftarrow l +$ SUMMAND $l_{t+1}(\theta, w, v, x, r)$
6 **return** l

PENALIZED-GRADIENT $\nabla l(N, \theta, w, v, x, r)$
INPUT: a parameter vector θ, starting values r_0, v_0, x_0
OUTPUT: the gradient $g \equiv \nabla l(\theta)$
1 **if** POSITIVITY-CONDITIONS$(\theta) \neq$ TRUE
2 $\quad$ **then return** PENALTY
3 $l \leftarrow$ SUMMAND $l_0(\theta, w, v, x, r)$
4 g $\leftarrow$ SUMMAND $\nabla l_0(\theta, w, v, x, r)$
5 **for** $t \leftarrow 0$ **to** $(N-1)$ **do**
6 $\quad l \leftarrow l +$ SUMMAND $l_{t+1}(\theta, w, v, x, r)$
7 $\quad$ g $\leftarrow$ g + SUMMAND $\nabla l_{t+1}(\theta, w, v, x, r)$
8 **return** g

The boolean expression POSITIVITY-CONDITIONS(θ) is TRUE $\iff$ $\theta \in \tilde{\Theta}$, and FALSE otherwise. Analogously to Rouah and Vainberg (2007), PENALTY is assumed constant and large enough to have the order of magnitude larger than the objective function evaluated at starting values.

4.3. **Results.**

	NM	FR	PR
$\hat{r_f}$	$= 9.000 \times 10^{-2}$	$= 9.000 \times 10^{-2}$	$= 9.000 \times 10^{-2}$
$\hat{\lambda}$	$2.800 \times 10^{+0}$	$= 1.800 \times 10^{+0}$	$= 1.800 \times 10^{+0}$
$\hat{n_x}$	$= 7.200 \times 10^{-6}$	7.673×10^{-6}	7.671×10^{-6}
$\hat{i_v}$	$= 9.000 \times 10^{-7}$	4.946×10^{-13}	3.834×10^{-14}
$\hat{i_x}$	$= 1.800 \times 10^{-6}$	2.786×10^{-6}	2.748×10^{-6}
$\hat{p_v}$	$= 5.400 \times 10^{-1}$	$= 5.400 \times 10^{-1}$	$= 5.400 \times 10^{-1}$
$\hat{p_x}$	$= 8.100 \times 10^{-1}$	$= 8.100 \times 10^{-1}$	$= 8.100 \times 10^{-1}$
$\hat{g_v}$	$= 3.600 \times 10^{+2}$	$= 3.600 \times 10^{+2}$	$= 3.600 \times 10^{+2}$
Time (s)	$0.0 \times 10^{+0}$	$1.3439 \times 10^{+2}$	$1.34516 \times 10^{+2}$
$\tilde{Q}_n(\hat{\theta})$	$-6.244133 \times 10^{+3}$	$-6.747140 \times 10^{+3}$	$-6.741407 \times 10^{+3}$

TABLE 2. The estimated coefficient values obtained in the estimation study using NM, FR and PR. PENALTY set to $999,999.999$. Symbol $=$ indicates that the values are equal to the initial values.

	SA	PG	SPG
$\hat{r_f}$	9.008×10^{-2}	$= 9.000 \times 10^{-2}$	9.003×10^{-2}
$\hat{\lambda}$	$1.800 \times 10^{+0}$	$= 1.800 \times 10^{+0}$	$= 1.800 \times 10^{+0}$
$\hat{n_x}$	3.917×10^{-5}	7.666×10^{-6}	5.381×10^{-5}
$\hat{i_v}$	1.079×10^{-6}	7.248×10^{-233}	4.167×10^{-6}
$\hat{i_x}$	3.702×10^{-6}	2.670×10^{-6}	2.960×10^{-7}
$\hat{p_v}$	5.398×10^{-1}	$= 5.400 \times 10^{-1}$	$= 5.400 \times 10^{-1}$
$\hat{p_x}$	8.102×10^{-1}	$= 8.100 \times 10^{-1}$	$= 8.100 \times 10^{-1}$
$\hat{g_v}$	$3.600 \times 10^{+2}$	$= 3.600 \times 10^{+2}$	$= 3.600 \times 10^{+2}$
Time (s)	1.72×10^{-1}	2.937×10^{0}	2.75×10^{0}
$\tilde{Q}_n(\hat{\theta})$	$-7.625983 \times 10^{+3}$	$-6.729235 \times 10^{+3}$	$-7.609727 \times 10^{+3}$

TABLE 3. The estimated coefficient values obtained in the estimation study using SA, PG and SPG. PENALTY set to $999,999.999$. Symbol = indicates that the values are equal to the initial values.

	FSQP-AL	FSQP-NL
$\hat{r_f}$	9.908070×10^{-2}	1.084674×10^{-1}
$\hat{\lambda}$	$1.557766 \times 10^{+1}$	$-9.986293 \times 10^{+1}$
$\hat{n_x}$	3.017787×10^{-5}	3.007165×10^{-5}
$\hat{i_v}$	3.068260×10^{-7}	1.509107×10^{-7}
$\hat{i_x}$	2.191491×10^{-6}	2.295927×10^{-6}
$\hat{p_v}$	2.525282×10^{-1}	1.805605×10^{-1}
$\hat{p_x}$	6.252131×10^{-1}	6.280134×10^{-1}
$\hat{g_v}$	$3.583449 \times 10^{+2}$	$3.623615 \times 10^{+2}$
Time (s)	1.88×10^{-1}	7.66×10^{-1}
$\tilde{Q}_n(\hat{\theta})$	$-8.432699 \times 10^{+3}$	$-8.435838 \times 10^{+3}$

TABLE 4. The estimated coefficient values obtained in the estimation study using FSQP.

We use the starting values reported in the third column (denoted "Estimation Starting Values") of Table 1. The results are reported in Tables 2 and 3. We notice that almost every unconstrained optimization algorithm performs unsatisfactorily. This may be due to the difficulty ensuring that the estimates remain in $\tilde{\Theta}$. The PENALTY encountered in this case introduces non-smoothness of the objective function, whereas many of the algorithms require smoothness for convergence. Similarly, Zivot (2009) reports that the GARCH log-likelihood function is not

always well behaved, which may cause difficulties for standard optimization techniques, especially when one takes into account the need to ensure that the positive variance and stationarity constraints hold.

Simulated annealing, as a local search algorithm (metaheuristic) capable of escaping local optima by allowing hill-climbing moves, fares better in our benchmark. As it is not gradient-based, smoothness is not an important requirement.

One may also notice that in some cases the only arguments changing significantly are n_x, i_v and i_x. This may result from the disproportionately high sensitivity of the objective function to those three arguments. In particular, compare the value of $\tilde{Q}_n$ for *Estimation Starting Values* in Table 1 to the one returned in case of the NM algorithm in Table 2 – a change in one variable (λ) is enough to significantly alter the value of the objective function. We suspect similar behavior in case of n_x, i_v and i_x for the FR, PR, PG, and SPG algorithms.

Below we present the summary of the estimation study results for the first seven algorithms:

(1) NM: only the value of λ has changed,
(2) FR: the value of i_x has improved compared to (1) (which probably led to an improvement in the objective function value), the i_v has worsened, long computation time,
(3) PR: very similar to FR, other than i_v is an order of magnitude further from the true value,
(4) BFGS: BFGS failed to produce results different from the starting values after $2.628280 \times 10^{+2}$ s,
(5) SA: better than all of the above,
(6) PG: exhibits a problem with i_v, other than that mediocre performance (on par with NM, FR, PR),
(7) SPG: nonmonotone line search combined with spectral choice of step length lead to a significant improvement compared to PG; i_v does not suffer problems as pronounced as in the case of FR, PR or PG, n_x is slightly better than in SA; still, the objective function value is slightly worse than SA. A possible reason: λ, p_v, p_x, g_v did not change at all.

Finally, we discuss the results in Table 4 for FSQP algorithm and consider two variants of it.

What is striking is that the objective function values have improved significantly compared to those obtained by the previously discussed

algorithms.[4] This suggests that FSQP might be the best choice for the estimation.

The estimate that is worth of attention is $\hat{\lambda}$. Estimation of this parameter appears difficult, possibly less so in the FSQP-AL case. However, the problem observed in all of the above algorithms, is that the estimates quite often remain equal to their initial values. This case is common in the component GARCH models – estimation issues were also reported by Christoffersen et al. (2008). Furthermore, it is worth noting that in the optimization settings we have used a very large interval – (-100, 100) – for the allowed values of the λ estimate compared to the true value. In practice, one would restrict it to an empirically reasonable (and realistic) range, depending on the beliefs and the experience of the researcher. For example, limiting it to (1, 3), we obtain an estimate of 1.72 by FSQP-AL and 1 by FSQP-NL (note the corner case). This seems to suggest that FSQP-AL deals better with this problem than FSQP-NL. We have also chosen to report the more negative results so as to not create an impression of an unfair treatment compared to the other algorithms.

As for the other results, the ones for r_f, n_x, i_x and g_v are comparable; FSQP-AL fares slightly better than FSQP-NL for i_v. Unfortunately, the estimates of p_v and p_x are not that close to the DGP ones (FSQP-AL fares slightly better for p_v). This might also be due to a small sample $N = 1,000$.

In order to find out whether rescaling the parameters to the same magnitude would yield an improvement, we simulate the model with DGP parameters set to r_f = 1.0e-001, λ = 5.0e-001, n_x = 4.0e-001, i_v = 1.0e-001, i_x = 2.0e-001, p_v = 2.5e-001, p_x = 7.5e-001, g_v = 1.0e+000. However, we experience similar difficulties as in the original case. Furthermore, one may argue that *ex ante* the researcher cannot always know the appropriate scale without performing estimation in the first place.

4.4. **Choices.** There are several directions one could investigate as far as estimation is concerned:

- using smoothly changing constraint-violation penalty function – for instance, as in CONT_CONSTRAINT in O2scl[5],

[4] In fact, they are slightly "better" than those corresponding to the DGP – we believe the reason is the finite sample of our simulation study and the effects of conditioning on the initial values not dying off.

[5] See `http://o2scl.sourceforge.net/o2scl/html/minimize_8h.html`.

- using constrained minimization algorithms allowing for constraints of type $\theta \in \tilde{\Theta}$ (which are not hypercubic), as in methods allowing for inequality constraints in Nocedal and Wright (2006),
- using the augmented Lagrangian method, see Conn et al. (1991),
- using constrained NLP, as in Altay-Salih et al. (2003).

Our solution is in the spirit of the last one, since we use specialized algorithms designed for optimization with inequality constraints, i.e. FSQP-AL and FSQP-NL. In addition, we have also considered Simulated Annealing, since it is reasonably fast and performs relatively well in the class of the unconstrained optimization algorithms.

5. Starting Values Selection Algorithm

As our next step, we consider a practical problem in model estimation: choosing the starting values. We consider an optimization method which might be very useful in practice. It belongs to the class of grid search methods, which means optimizing without choosing the initial parameter values.

First, we consider the naive grid generation, using unconditionally uniformly distributed initial values and box constraints for the support: $r_f \in \mathbb{R}$, $\lambda \in \mathbb{R}$, $n_x > 0$, $i_v > 0$, $p_v \in (0,1)$, $p_x \in (0,1)$, $g_v \in \mathbb{R}$. We find that the naive grid search method performs very poorly in practice. Generating feasible starting values this way takes a very significant amount of time ($>$ 1.0e+3 s). Even though the FSQP algorithm we use allows for non-feasible starting values and begins by searching for feasible ones, it takes too much time to use this method in practice. None of the algorithms was able to get close to the results obtained with the starting values in Table 1.

Therefore, we introduce a grid search method which we shall refer to as the Conditionally-Uniform Feasible (CUF) Grid Search (CUFGS). The idea behind it is to solve the main weakness of the naive method: the unacceptably long amount of time it takes to generate feasible initial values. The way to solve this problem is to generate the starting values sequentially, starting by obtaining the parameters which are not affected by inequality constraints by drawing their values from a uniform distribution. Next, the constrained parameters are generated conditioning on the ones associated with their constraints. The choice of the conditional distributions ensures that the values of the generated parameters satisfy the inequality constraints by definition. The algorithm is as follows:

GENERATE-CUF-PARAMETERS
INPUT: lower (bl) and upper (bu) box bounds, inequality constraints
OUTPUT: the set of CUF parameters satisfying positivity conditions $\theta \in \tilde{\Theta}$

0 **Generate** $r_f \sim U(bl_{r_f}, bu_{r_f}), \lambda \sim U(bl_\lambda, bu_\lambda)$
1 **do**
2 **Generate** $i_v \sim U(bl_{i_v}, bu_{i_v}), i_x \sim U(bl_{i_x}, bu_{i_x}), g_v \sim U(bl_{g_v}, bu_{g_v})$
3 $bl_{p_v} \leftarrow i_v g_v^2$ (feasible lower bound for p_v)
4 $bl_{n_x} \leftarrow i_v + i_x$ (feasible lower bound for n_x)
5 **while** $bl_{p_v} > bu_{p_v}$ **OR** $bl_{n_x} > bu_{n_x}$
6 **Generate** $n_x \sim U(bl_{n_x}, bu_{n_x}), p_v \sim U(bl_{p_v}, bu_{p_v}), p_x \sim U(p_v, bu_{p_x})$
7 **return** $\theta := (r_f,\ \lambda,\ n_x,\ i_v,\ i_x,\ p_v,\ p_x,\ g_v)^\mathsf{T}$

This procedure ensures that the generated initial values satisfy (2.8)-(2.10). It is also much faster than searching for the feasible region by optimization, as it only relies on the very fast (conditionally-)uniform pseudo-random number generator.

Next, we perform a CUF Grid Search a given number of times. Each time we obtain the starting values by using GENERATE-CUF-PARAMETERS and passing them to the optimization algorithm. We keep track of the best objective function value achieved in each iteration. Finally, we return the overall best result. This allows us to optimize by only specifying (very rough) lower and upper bounds for the parameters, instead of manually choosing the starting values.

Table 5 contains a summary of the CUFGS estimation study results. The best objective function value achieved and the number of iterations as well as the computation time in seconds are reported.[6]

Algorithm	$\tilde{Q}_n$	Iterations	Time (s)
NM	$-4,477.69$	$3,370$	56.829
FR	$4,283.83$	19	167.422
PR	$4,283.83$	19	167.016
BFGS	-366.88	20	327.047
SA	$-3,505.79$	$2,531$	776.531
PG	$-4,336.41$	203	305.953
SPG	$-3,089.56$	17	553.422
FSQP-AL	$-8,433.92$	$1,740$	47.218
FSQP-NL	$-8,434.50$	503	20.250

TABLE 5. The estimated coefficient values obtained in the estimation study using CUFGS. PENALTY set to $999,999.999$.

[6]To make the comparison practically interesting, we have imposed an upper limit of 1,000 s for the grid search.

It is seen from the table that the difference between the specialized and non-specialized algorithms is quite pronounced. Only the FSQP manages to achieve the results comparable to those obtained when the optimization process is started from manually-chosen starting values close to the DGP ones. It is also interesting to note the trade-off between the convergence properties (number of iterations) and the computational cost of each iteration. It is not always the case that the algorithms with faster convergence properties (a lower number of necessary iterations) perform best, since each iteration may be sufficiently costly to offset any gains in accuracy.

The results for the FSQP CUFGS optimization are shown in more detail in Table 6. We note that r_f estimates are comparable for both FSQP-AL and FSQP-NL, n_x and i_v are estimated slightly better by FSQP-AL, in the case of i_x FSQP-AL is much closer to DGP than FSQP-NL, for p_v FSQP-AL gets very close to DGP, whereas in the case of p_x FSQP-NL is quite close to DGP. Finally, the estimates of λ and g_v are not as good as the ones resulting from manually chosen starting values that were close to the DGP. Our conclusion is that it is worth to try both FSQP-AL and FSQP-NL and to experiment with the choice of lower and upper bounds for λ and g_v. Prior experience with the estimation of those two parameters might lead to a choice of the narrower bounds containing only "reasonable" values, which then leads to good parameter estimates. In our case we have allowed the λ to vary from -10 to 10 and g_v to from -1,000 to 1,000.

We also note that some of the final iterations in both cases (AL and NL) lead to numerically close objective function values (indicating possible flatness of the objective function), so looking at several sets of the estimates (instead of only the best ones) might also be advisable if the values are sufficiently close to being numerically indistinguishable.

6. Estimation Results

As a practical example, we use estimation of the SCGARCH model of Dziubinski (2011). Due to results in the previous section we have chosen FSQP and SA to estimate the model parameters. In this section we provide a brief summary of the methodology and the results from the numerical perspective and refer the reader to Dziubinski (2011) for further details.

	FSQP-AL	FSQP-NL
$\hat{r_f}$	1.012×10^{-1}	1.012×10^{-1}
$\hat{\lambda}$	$-1.000 \times 10^{+1}$	$-1.000 \times 10^{+1}$
$\hat{n_x}$	2.718×10^{-5}	8.294×10^{-8}
$\hat{i_v}$	2.189×10^{-6}	5.399×10^{-8}
$\hat{i_x}$	8.009×10^{-7}	4.307×10^{-37}
$\hat{p_v}$	6.594×10^{-1}	9.921×10^{-1}
$\hat{p_x}$	6.625×10^{-1}	9.990×10^{-1}
$\hat{g_v}$	$-3.523 \times 10^{+0}$	$-9.986 \times 10^{+2}$
Time (s)	$4.7218 \times 10^{+1}$	$2.025 \times 10^{+1}$
$\tilde{Q}_n(\hat{\theta})$	$-8.43392 \times 10^{+3}$	$-8.4345 \times 10^{+3}$

TABLE 6. The estimated coefficient values obtained in the estimation study using FSQP CUFGS.

For the purposes of research reproducibility, we report the starting values in column "Estimation Starting Values" in Table 1.[7] We restart the optimization using 100, 1,000, 10,000 and 50,000 iterations, consecutively – we find the lowest values of $\tilde{Q}_n$ with 10,000 iterations in case of both the daily and the 5-minute data.

The estimates obtained using the SA optimization algorithm are quite stable over the sampling frequency. One of the issues, however, is that some of the coefficients are statistically insignificant as their standard errors are large. Because of that, we do not put trust in those results – also since there are several scaling issues when using the SA algorithm for optimization; we note, that the objective function values are inferior to those obtained using FSQP.

Next, we consider the estimates obtained using FSQP-AL CUFGS optimization algorithm. The objective function values improve and large standard errors (indicating insignificance) are practically not encountered in FSQP optimization. This strengthens our belief that the choice of the optimization algorithm matters a great deal in this regard.

The results for the FSQP-NL CUFGS optimization algorithm are mostly similar to those discussed above, except that in this case the estimate of g_v is better.

In order to examine the scaling issues with the SA algorithm, we also perform the optimization with rescaling. The procedure is as follows:

[7]Note, that as Zivot (2009) reports, poor choice of starting values can lead to an ill-behaved log-likelihood and cause convergence problems – this is why we use the starting values that satisfy the non-negativity conditions.

(1) rescale the θ in objective function computations, using the orders of magnitude from the FSQP CUFGS results (note that this requires prior estimation using FSQP CUFGS),
(2) use starting values equal to 1.0 for all coefficients,
(3) perform SA CUFGS optimization (grid search is necessary, since we find that rescaled non-grid optimization fails to converge with given starting values),
(4) store the results of the SA CUFGS optimization,
(5) use the stored results in non-scaled non-grid SA optimization to obtain final results.

We note the improvements in the objective function value in case of low-frequency data. However, we would generally advise against this estimation method compared to FSQP CUFGS, since:

(1) prior estimation using different algorithm (such as FSQP CUFGS) is still necessary for this method – we find SA to be very sensitive to scaling,
(2) it is impractically slow – not only due to prior estimation requirements, but also because SA CUFGS takes a considerable amount of time to find feasible solutions ($>$ 5 min for low-frequency data, $>$ 20 min for high-frequency data) and then to improve upon them in the final step ($>$ 2.5 min, $>$ 10 min, respectively). In comparison, FSQP-AL needed $<$ 4 min to converge for high-frequency data.

7. Conclusions

In this paper we consider a collection of numerical methods pertinent to the estimation of the SCGARCH model of Dziubinski (2011). We discuss and provide a benchmark of several optimization routines applied to this estimation problem. We find that the FSQP algorithms (FSQP-AL and FSQP-NL) have the best performance of the algorithms we consider.

We also provide an algorithm that can be profitably used in the selection of starting values for the estimation. We find that the FSQP algorithms coupled with CUFGS perform reasonably well and significantly better than the non-specialized optimization algorithms.

We believe that our algorithm is applicable to a wide-range of nonlinearly constrained optimization problems, where there is a sequential functional dependence among the constraints on the variables. The algorithm is especially useful in practical econometric modeling, where the choice of starting values is often not an obvious one, especially for the end-users unfamiliar with the given modeling framework, or where

the model is new and there is no well-known range of parameters' bounds.

References

Altay-Salih, A., M. C. Pinar, and S. Leyffer (2003): "Constrained Nonlinear Programming for Volatility Estimation with GARCH Models," *SIAM Review*, 45, 485–503.

Birgin, E. G., J. Mario, and M. M. Raydan (2000): "Nonmonotone spectral projected gradient methods on convex sets," *SIAM Journal on Optimization*, 10, 1196–1211.

Brooks, C., S. P. Burke, and G. Persand (2001): "Benchmarks and the accuracy of GARCH model estimation," *International Journal of Forecasting*, 17, 45 – 56.

Christoffersen, P., K. Jacobs, C. Ornthanalai, and Y. Wang (2008): "Option valuation with long-run and short-run volatility components," *Journal of Financial Economics*, 90, 272–297.

Conn, A. R., N. I. M. Gould, and P. Toint (1991): "A Globally Convergent Augmented Lagrangian Algorithm for Optimization with General Constraints and Simple Bounds," *Siam Journal on Numerical Analysis*, 28.

Cormen, T. H., C. E. Leiserson, R. L. Rivest, and C. Stein (2001): *Introduction to Algorithms*, McGraw-Hill Science / Engineering / Math, 2nd ed.

Dreo, J., A. Pétrowski, P. Siarry, and E. Taillard (2005): *Metaheuristics for Hard Optimization: Methods and Case Studies*, Springer.

Dziubinski, M. P. (2011): "Option Valuation with the Simplified Component GARCH (SCGARCH) Model," *CREATES Research Paper 2011-9*.

Henderson, D., S. H. Jacobson, and A. W. Johnson (2003): "The Theory and Practice of Simulated Annealing," in *Handbook of Metaheuristics*, ed. by F. Glover and G. A. Kochenberger, Kluwer Academic Publishers, International Series in Operations Research & Management Science, chap. 10, 287–320.

Kelley, C. T. (1999): "The Gradient Projection Algorithm," in *Iterative Methods for Optimization*, SIAM, chap. 5.4, 91–96.

Lawrence, C., J. L. Zhou, and A. L. Tits (1997): "User's Guide for CFSQP Version 2.5: A C Code for Solving (Large Scale) Constrained Nonlinear (Minimax) Optimization Problems, Generating Iterates Satisfying All Inequality Constraints," Tech. rep., Institute for Systems Research, University of Maryland, Technical Report TR-94-16r1.

McCullough, B. D. and C. G. Renfro (1999): "Benchmarks and software standards: A case study of GARCH procedures," *Journal of Economic and Social Measurement*, 25, 59–71.

Nocedal, J. and S. Wright (2006): *Numerical optimization*, Springer: Springer, 2. ed. ed.

Rouah, F. D. and G. Vainberg (2007): *Option Pricing Models and Volatility Using Excel-VBA*, Wiley Publishing.

Zivot, E. (2009): "Practical Issues in the Analysis of Univariate GARCH Models," in *Handbook of Financial Time Series*, Springer, 113–155.

(Matt P. Dziubinski) CREATES
School of Economics and Management
University of Aarhus
Building 1322, DK-8000 Aarhus C
Denmark

E-mail address, Matt P. Dziubinski: matt@creates.au.dk

Research Papers 2012

CREATES
Center for Research in Econometric Analysis of Time Series

2011-40: Søren Johansen and Bent Nielsen: Asymptotic theory for iterated one-step Huber-skip estimators

2011-41: Luc Bauwens, Arnaud Dufays and Jeroen V.K. Rombouts: Marginal Likelihood for Markov-switching and Change-point Garch Models

2011-42: Manuel Lukas: Utility-based Forecast Evaluation with Multiple Decision Rules and a New Maxmin Rule

2011-43: Peter Christoffersen, Ruslan Goyenko, Kris Jacobs, Mehdi Karoui: Illiquidity Premia in the Equity Options Market

2011-44: Diego Amaya, Peter Christoffersen, Kris Jacobs and Aurelio Vasquez: Do Realized Skewness and Kurtosis Predict the Cross-Section of Equity Returns?

2011-45: Peter Christoffersen and Hugues Langlois: The Joint Dynamics of Equity Market Factors

2011-46: Peter Christoffersen, Kris Jacobs and Bo Young Chang: Forecasting with Option Implied Information

2011-47: Kim Christensen and Mark Podolskij: Asymptotic theory of range-based multipower variation

2011-48: Christian M. Dahl, Daniel le Maire and Jakob R. Munch: Wage Dispersion and Decentralization of Wage Bargaining

2011-49: Torben G. Andersen, Oleg Bondarenko and Maria T. Gonzalez-Perez: Coherent Model-Free Implied Volatility: A Corridor Fix for High-Frequency VIX

2011-50: Torben G. Andersen and Oleg Bondarenko: VPIN and the Flash Crash

2011-51: Tim Bollerslev, Daniela Osterrieder, Natalia Sizova and George Tauchen: Risk and Return: Long-Run Relationships, Fractional Cointegration, and Return Predictability

2011-52: Lars Stentoft: What we can learn from pricing 139,879 Individual Stock Options

2011-53: Kim Christensen, Mark Podolskij and Mathias Vetter: On covariation estimation for multivariate continuous Itô semimartingales with noise in non-synchronous observation schemes

2012-01: Matei Demetrescu and Robinson Kruse: The Power of Unit Root Tests Against Nonlinear Local Alternatives

2012-02: Matias D. Cattaneo, Michael Jansson and Whitney K. Newey: Alternative Asymptotics and the Partially Linear Model with Many Regressors

2012-03: Matt P. Dziubinski: Conditionally-Uniform Feasible Grid Search Algorithm