Notes on GAUSS programs used for *Avoiding the pitfalls ...* by van Norden & Vigfusson

Simon van Norden, April 1998

1.0 Quick Start

1. Run GAUSS 3.2 with the Maxlik() version 4.0 module under Windows95 on an Intel-compatible processor.
2. Dump all the files into *c:\gauss\pitfalls.*
3. Run one of the **.gau* files.

2.0 Introduction

This file documents the basics of the GAUSS programs used to produce the results reported in the above-mentioned paper, forthcoming in *SNDE*. There are several qualifications that should be noted at the outset.

- the programs are incomplete. The programs used for the power calculations in Section 4 are not included, nor are subsidiary programs used to construct the measures of deviations from fundamentals used in the van Norden test. However, the programs included here should enable users to reproduce most of the results from Section 3 on test size. In particular, they include our data set and the underlying code used to estimate regime-switching models. These should allow users to apply these tests to data sets of their choice and perform their own Monte Carlo experiments.
- while the programs are accurate so far as we know, we don't legally guarantee their suitability for any particular purpose. If you find what you think are errors in the programs, we would appreciate hearing from you. We also make no commitment to keep this code up to date. Software platforms change and our time is limited. This code works with our current version of GAUSS (3.2.29, May 1997); please don't ask us to adapt it to older or newer versions.
- these instructions are no substitute for a good knowledge of GAUSS. If you are new to GAUSS and are having trouble getting things to run, *Read The Manual First (RTFM)!*
- these programs will not run on a non-Intel platform. The Markov-switching procedures using externally compiled C-code to speed evaluation; this binary code is platform dependent. In addition, many binary GAUSS matrix-format files are included; these would need to be converted if used on other platforms using the *transdat* utility Aptech includes with other releases of GAUSS for this purpose.

Having said all that, the files included fall into 5 groups.

1. GAUSS library and procedures (including external C-code.)
2. Data files
3. GAUSS programs for performing bubble tests on historical data (and some associated parameter values saved in matrix files.)
4. GAUSS programs for Monte Carlo simulations
5. Additional utilities and programs for data description.

Let's look at each of these in turn.

3.0 *GAUSS* Library and Procedures

The basic code that defines the likelihood function for regime-switching models, their gradients, the smoothed probabilities and the p-value graphs are coded as GAUSS procedures. This is a slightly updated version of the GAUSS code documented in Bank of Canada working paper 96-3 by van Norden and Vigfusson and downloadable from *www.bank-banque-canada.ca*.[1]

We assume that you run GAUSS from *c:\gauss* and that you have unzipped all our files into *c:\gauss\pitfalls*. Our programs therefore include *pitfalls.lcg* in the *library* statement at the top of the file so that GAUSS knows where to find pre-defined procedures. If you are getting "File not found" errors, this is probably the source of the problem. Check the path to the *pitfalls.lcg* file in your library statement, and check the paths in the *pitfalls.lcg* file itself. Programs estimating Markov-switching models (such as for the Hall & Sola test) include the *dlibrary* statement to tell GAUSS where it can find the external C-code *swvv.dll*.

Our library file references the following procedures:

TABLE 1. Files and Procedures References in *Pitfalls.lcg*

File	Procedures	Notes
swlf.g	swli swgrad	Likelihood function and Gradients for simple switching regressions
swcontam.g	swecli swecgrad	Likelihood function and Gradients when only variances differ (Beta1=Beta2)
swinit2.g	selif binitial	Initial values for switching regression
swintcon.g	becinit	Initial values for switching regression when only variances differ (Beta1=Beta2)

1. For those of you familar with the former, no, we still have not coded the EM algorithm for time-varying transition probabilities in the Markov model.

TABLE 1. Files and Procedures References in *Pitfalls.lcg*

File	Procedures	Notes
tedec.dec	_emmaxit : matrix _emconvg : matrix _emecm : matrix _emprint : matrix	Globals for control of EM estimation (simple switching)
swem.g	swem	Single iteration of EM
swecmem.g	swecmem	Single iteration of EM when only variances differ (Beta1=Beta2)
swemmle.g	swemmle	Control Program for EM with integrated maxlik call
mcswemmle.g	mcsw	Modified version of above for Monte Carlos (reduced output)
swdescrp.g	swdescrp	Descriptive Statistics for estimated switching regression
swcascad.g	maxrprt swcascad	Cascading estimation of switching regression models
newcascd.g	newcascade	Suite of Misspecification tests for switching regressions
swmisspc.g	swmisspc	
lfmkvDLL.g	mkvstart swmkv	Likelihood Function and Starting Values (Markov Switching Regression)
angrad.g	angrad	Gradients (Markov Switching Regression)
kimsmthv.g	kimsmth	Smoother
emsmthv.g	emsmth	Smoother for the EM algorithm
emmarkov.g	incdfn em	EM Algorithm for Markov Switching Regressions (Constant Transition probabilities only!)
mkvspec.g	mkvspec	Suite of Misspecification tests for Markov switching regressions
wmisspec.g	wmisspec	White's Misspecification Test
bihist.g	bihist	Bivariate (i.e. 3-D) Histograms. Also see MaxHist().
pvalplot.g	pvalplot pval_chi2 pval_normal pval_f cv_ks	Davidson & MacKinnon p-Value plots
shadebox.g	shadebox	Shade rectangles on GAUSS graphs
mkvshufl.g	shuffles HallShfl	Shuffles() is a simple randomization proc that accepts a seed.
hallstat.g	hallstat	Hallstat() constructs and returns the Wald and the two t-stats.
shuffle.g	shuffle scascade waldshuf	waldshuf can take the parameter estimates and spit out the Wald stat for the van Norden test.

4.0 Data Files

There are four data files in ASCII format. These files start with one line containing the names of the series; each subsequent line has one observation on each variable.

TABLE 2. Data files

Name	Contents	Notes
tseindex.asc	TSE 300 index	For Hall & Sola Test
sp5index.asc	S&P 500 index (607 obs. from 1947M1 to 1997M7)	For Hall & Sola Test
bubsp500.asc	The series covers the period 1962M9 to 1997M5	For van Norden Test
	dpsp500: Change in log S&P500 index (not used)	
	rsp500: Excess returns on S&P500 index	
	bsp500cr: Bubble using cointegrating regression (not used)	
	bsp500cs: Bubble using Campbell-Shiller model	
bubtse.asc	The series covers the period 1956M11 to 1997M6	For van Norden Test
	dptse: Change in log S&P500 index (not used)	
	rtse: Excess returns on S&P500 index	
	btsecr: Bubble using cointegrating regression (not used)	
	btsecs10: Bubble using Campbell-Shiller model	

In addition, there are several files in a binary GAUSS matrix (*.*fmt*) format.[1] These are generally used to store parameter vectors that are used for starting values. These will be described below.

5.0 Program Files for Historical Data

The van Norden test is estimated by *btse.gau* and *bsp500.gau* for the TSE and S&P500 data sets. Results are stored in *.*out* and estimated parameter vectors are in **cs.fmt*. The Hall & Sola test is estimated by *bsp5hall.gau* and *btsehall.gau* for the S&P500 and the TSE data sets. Results are again stored in *.*out* and estimated parameters are in *.*fmt*.[2]

1. Note that, like all the files in this archive, the binary GAUSS matrix file is in GAUSS' Intel format; if you are using GAUSS on a non-Intel (e.g. workstation or Mac) platform, you may need to convert this using the *transdat* utility that comes with your version of GAUSS.
2. As currently configured, they use the *.*fmt* files for initial estimates of the parameter vectors so convergence is extremely rapid. However, users can substitute other starting estimates to check robustness.

6.0 Program Files for Bootstrapping

6.1 van Norden Test

The bootstrap results for the van Norden test are done in two steps. The basic principle is that the measures of the bubble are reordered independently of the excess returns series, thereby destroying any relationship between the two and allowing us to examine the distribution of our statistics under this null. We are interested in the Wald test that $\beta_S = \beta_C$ and the LR tests of the NM and VR models. Since the NM and VR models contain no measure of the bubble, their fit will be unaffected by the randomization experiment so we need not re-estimate them. The values of their log-likelihood functions are stored in *LNMTSE.fmt, LNMSP500.fmt, LVRTSE.fmt, and LVRSP500.fmt* while their parameter estimates are in *BNMTSE.fmt, BNMSP500.fmt, BVRTSE.fmt, and BVRSP500.fmt.*

The main program which bootstraps the data and re-estimates the model is *shuffle.gau*. It must be modified by hand to use either the S&P500 or the TSE data.[1] On each iteration, this program reorders the bubble series, re-estimates the switching model, opens the output file, appends one line of results from the latest iteration, then closes the output file. This allows the program to be interrupted without losing the results done to that point. Note that the reordering is done using a seeded random-number generator so that our results should be as reproduceable as possible.

Shuffle.gau writes out six figures per line

- the LR test of EC vs SWP
- the LR test of ECP vs SWP
- the LR test of NMP vs SWP
- the Wald test for $\beta_0 = \beta_0$
- the Wald test for $\beta_C = \beta_S$
- a return code (=0 if everything went well.)

6.2 Hall & Sola Test

The bootstrap results for the Hall & Sola test are done in much the same way as those for the van Norden test. In this case, the change in log prices are randomly reordered, thereby destroying any temporal dependence and allowing us to examine the distribution of our statistics under this null. We are interested in the Wald test that $\beta_S = \beta_C$, and use estimates of the NM model (from *bnmhssp5.fmt* and *bnmhstse.fmt*) as initial parameter estimates.

1. The only required changes are (1) the name of the output file, (2) the name of the data file, and (3) the names of files from which the parameter vectors for the NM and VR models are read.

The main program which bootstraps the data and re-estimates the model is *mkvshufl.gau*. It must be modified by hand to use either the S&P500 or the TSE data.[1] On each iteration, this program resimulates the price series, re-estimates the markov switching model, opens the output file, appends one line of results from the latest iteration, then closes the output file. This allows the program to be interrupted without losing the results done to that point. *mkvshufl.gau* writes out seven figures per line; The iteration number, the elapsed time, the two betas, their two variances and their covariance. Results are converted to test statistics using the *hallstat.g* procedure, which is called in *sizegraf.gau* (described below.)

We found that convergence and estimation of the VCV matrix was a much greater problem with these models than with the van Norden test. For that reason, we introduced a number of changes into the program. First, instead of seeding the random number generator once at the start of the program, we used a matrix of 2500 random integers *seeds.fmt* to seed the random re-ordering for each iteration. This allows us to return to those cases were estimation of the VCV matrix failed (indicated by a row of zeros in the output file) and re-estimate them using more computationally-intensive methods.

7.0 Utilities for Data Description

The tabulation of the Monte Carlo results is done in *sizegraf.gau*, which also constructs the p-value and p-value discrepency plots.

1. The only required changes are (1) the name of the output file, (2) the name of the data file, and (3) the names of files from which the parameter vectors for the NM and VR models are read.