Solving, Estimating and Selecting Nonlinear Dynamic Economic Models without the Curse of Dimensionality

Viktor Winschel[1]

July 29, 2005

Abstract: A welfare analysis of a risky policy is impossible within a linear or linearized model and its certainty equivalence property. The presented algorithms are designed as a toolbox for a general model class. The computational challenges are considerable and I concentrate on the numerics and statistics for a simple model of dynamic consumption and labor choice. I calculate the optimal policy and estimate the posterior density of structural parameters and the marginal likelihood within a nonlinear state space model. My approach is even in an interpreted language twenty time faster than the only alternative compiled approach. The model is estimated on simulated data in order to test the routines against known true parameters. The policy function is approximated by Smolyak Chebyshev polynomials and the rational expectation integral by Smolyak Gaussian quadrature. The Smolyak operator is used to extend univariate approximation and integration operators to many dimensions. It reduces the curse of dimensionality from exponential to polynomial growth. The likelihood integrals are evaluated by a Gaussian quadrature and Gaussian quadrature particle filter. The bootstrap or sequential importance resampling particle filter is used as an accuracy benchmark. The posterior is estimated by the Gaussian filter and a Metropolis-Hastings algorithm. I propose a genetic extension of the standard Metropolis-Hastings algorithm by parallel random walk sequences. This improves the robustness of start values and the global maximization properties. Moreover it simplifies a cluster implementation and the random walk variances decision is reduced to only two parameters so that almost no trial sequences are needed. Finally the marginal likelihood is calculated as a criterion for nonnested and quasi-true models in order to select between the nonlinear estimates and a first order perturbation solution combined with the Kalman filter.

Keywords: welfare analysis, certainty equivalence, stochastic dynamic general equilibrium model, statistical decision theory, Bayesian econometrics, Smolyak operator, high dimensional approximation and integration, quadrature, Chebyshev polynomials, nonlinear state space model, Kalman filter, Gaussian filter, Gaussian particle filter, sequential importance resampling filter, posterior density of structural parameters, genetic Metropolis-Hastings algorithm, global maximization, marginal likelihood, variable of interest

JEL classification: E0, F0, C11, C13, C15, C32, C44, C52, C63, C68, C88

[1]**Address:** University of Mannheim, Department of Economics, L7,3-5, 68131 Mannheim, Germany; phone: +49/621/181-1817, fax: +49/621/181-1814, email: viktor_winschel@web.de

Contents

1 Introduction

Linear or linearized models are a drawback for policy evaluation and recommendation. They cannot account for important welfare channels under certainty equivalence when risk aversion and thus higher moments matter. Certainty equivalence describes the property of a policy to be independent of shock variances. Asset pricing models relate prices to risk measured by variances. The optimal currency area literature identifies the relative shock size as an important criterium for countries to join a currency union. The closely related theory of optimal exchange rate regimes discusses the optimal trade off between risks concerning prices, exchange rates or business cycles. It shapes the optimal policy as well as historical policy evaluations. The new open economics initiated by Obstfeld and Rogoff (1995) or the modern monetary economics, for example in Walsh (2001), take the nonlinearities into account when deriving optimal policies. In positive economics nonlinear models help for example to account for business cycles features as in Morley and Piger (2005). My work provides a Bayesian estimation framework for general nonlinear macroeconomic models with rational expectations.

The estimation consists of three main steps. The first step is to solve the model and the second to evaluate the implied likelihood. The last step is a Metropolis-Hastings algorithm which generates a sample from the posterior density of structural parameters. The model can be linear or not, with rational expectations or more general learning processes, Gaussian distributed shocks or not, observed and unobserved variables, fixed or time varying parameters. A model selection criterion selects parameter estimates whether models are nested or not, true or quasi-true. The generated sample from the posterior density can be used for a density estimate of a variable of interest. This can be any nonlinear function of observables and unobservables like a specification test statistic or any other economic variable. Most interesting are predictions and welfare measures for future or counterfactual policy simulations.

Beside the statistical difficulties to estimate a density, economic models are difficult to solve. The problem is to derive the optimal policy from a nonlinear functional equation with expectations. Approximating the solution of a model is now well understood and documented in books like Judd (1998) or Marimon and Scott (1999). An applied solution approach with mixed discrete and continuous variables can be found in Miranda and Fackler (2002). Aruoba, Fernádez-Villaverde, and Rubio-Ramírez (2003) have recently compared the accuracy of available approaches for the same model as I use in the following.

Extensions of the nonlinear solution methods towards a likelihood based estimation are rare and the only one I know can be found in Fernádez-Villaverde and Rubio-Ramírez (2004b). They approximate the model solution by finite elements, rational expectations are evaluated by Kronecker Gaussian quadrature, the likelihood of the implied nonlinear state space model is calculated by a boot-

strap filter and the Metropolis-Hastings algorithm with random walk innovations is used to draw random vectors from the posterior of structural parameters. This approach is very general and can handle any nonlinearity and shock distribution of the model. In particular nonsmooth policy functions can be approximated which may result from inequality constraints or min and max functions in the model formulation. The most urgent problem is that even with Fortran code they need around four days to estimate the smallest possible model and computationally more efficient methods are needed. The approach I present is related to their framework and extends and varies it in several dimensions with emphasis on faster algorithms for a general class of models.

The contribution of this work is to introduce the Smolyak operator for approximation and integration in econometrics, to extend the Metropolis-Hastings algorithm, to give an overview, implement and compare some available nonlinear state space filters and develop a software for the estimation of a general nonlinear model class.

The asymptotic convergence analysis suggests that Monte Carlo integration does not exploit smoothness of the integrand. If smoothness of a function is defined by the number of finite derivatives then economic models are often characterized by infinitely smooth functions. Quadrature based deterministic approaches have convergence rates which depend on the ratio between dimension and smoothness, see for example in Gerstner and Griebel (2003). This permits to trade off dimension against smoothness. The Smolyak operator extends approximation and integration from the univariate setting to the multivariate one. It relieves substantially the curse of dimensionality by modifying the usual Kronecker product. Integration is needed for rational expectations, weighted residuals, state space filters, posterior of structural parameters and marginal likelihoods. An approximation is needed to obtain the optimal policy function. The operator applies for a spectral as well as a finite element approximation and it works behind the scenes as complete polynomials in the perturbation approach. Moreover it could be of interest for nonparametric econometrics where the domain the involved variables is divided into several subdomains.

The proposed genetic extension of the standard random walk Metropolis-Hastings algorithm allows for an unbiased convergence test and an automated choice of the optimal variance for the candidate parameter vectors. It simplifies a cluster implementation of the estimation and improves the global maximization properties of the algorithm.

Since the likelihood evaluation is the main bottleneck I implement a Smolyak based Gaussian quadrature filter in order to exploit the model smoothness. The Gaussian filter is compared to the most simple but computationally expensive bootstrap filter which is also called sequential importance resampling particle filter.

The implementation for a general model class serves the purpose to solve and estimate a model without adjusting the code. Only the first order conditions and

some control parameters of the algorithms have to be provided. The disadvantage of a general approach is that special features of a model cannot be exploited to accelerate the estimation algorithms. On the other hand efforts to improve the algorithmic efficiency can be devoted to one single code.
The simplest alternative estimation technique is GMM. It uses only the moments implied by the first order conditions instead of the whole information available through the solution and the implied dynamics of the model. This information is embedded in the likelihood. It is a complete density of the observed variables and not only some of their moments. Waste of information results in poor small sample properties and is therefore problematic in usual macroeconomic samples of 100 or even less observations. Moreover GMM is not suited for the model selection and a likelihood based approach is indispensable for this purpose. The popularity of GMM is related to its simple implementation where structural parameters can be estimated without solving the model and evaluating its likelihood.
A linear estimator uses a first order Taylor expansion of the first order conditions around the deterministic steady state. The policy function is obtained from the implied quadratic matrix or Riccati equation by a generalized Schur decomposition. The estimator finally uses a likelihood evaluation techniques like the Kalman filter for the involved linear Gaussian state space model. In addition to biased estimates, this procedure imposes certainty equivalence and is thereby likely to miss important quantitative welfare relationships. I use this linear estimation approach to compare it with the nonlinear estimates.
Section 2 gives an overview of the econometric problems to estimate the posterior density and the marginal likelihood. Section 3 describes the solution method and the Smolyak operator. Section 4 presents the general principle of filtering and various approaches to evaluate the likelihood of a nonlinear state space model. Section 5 summarizes the Metropolis-Hastings algorithm and the proposed genetic extension. Section 7 presents a Monte Carlo simulation study to evaluate the performance of the genetic extension. It reports the estimated parameters and the marginal likelihood of the nonlinear and linearized model. Section 8 shortly describes the developed software and section 9.

2 Econom(etr)ics

The principle of information accounting in Bayesian econometrics is the same as in positive and normative economics and Geweke (2005) writes:

> *The strategic advantage of Bayesian statistics stems from the fact that its conditioning is driven by the actual availability of information, and its complete integration with the theory of economic behavior under uncertainty, achieved by Friedman and Savage (1948) and Friedman and Savage (1952).*

Hansen and Heckman (1996) write:

> *The rational agents in real business cycle models use this [statistical decision] theory and, as a consequence, are assumed to process information in a highly structured way. Why should the producers of estimates for the real business cycle models act differently?*

A decision unit in the statistical decision theory is defined by the utility function, variable of interest, policy, models and prior information $\mathcal{D} = \{ U, \boldsymbol{\omega}, \boldsymbol{x}\ M, p \}$. Utility $U(\boldsymbol{\omega}, \boldsymbol{x})$ is obtained from the variable of interest $\boldsymbol{\omega}$ and the policy $\boldsymbol{x}$.[2] A decision is usually assisted by alternative models $M = \{M_1, ...M_m\}$. A model M_i is represented by the likelihood or density of observables given unobservables $p(\boldsymbol{y} \,|\, \boldsymbol{\theta}_{M_i}, M_i)$. Observables $\boldsymbol{y}$ and their realizations $\boldsymbol{y}^0$ are explained by unobservables $\boldsymbol{\theta}_{M_i}$. Unobservables are structural parameters $\boldsymbol{\theta}^p_{M_i} \subset \boldsymbol{\theta}_{M_i}$ and states or latent variables $\boldsymbol{s}_{M_i} \subset \boldsymbol{\theta}_{M_i}$. Prior information is encoded in the prior density of unobservables $p(\boldsymbol{\theta}_{M_i} | M_i)$ and the prior probabilities of models $p(M_1), ... p(M_m)$. The variable of interest is any function $p(\boldsymbol{\omega} | \boldsymbol{y}, \boldsymbol{\theta}_{M_i}, M_i)$. Competing models may use different unobservables $\boldsymbol{s}_{M_i} \neq \boldsymbol{s}_{M_j}$ to explain the same observables $\boldsymbol{y}_{M_i} = \boldsymbol{y}_{M_j}$. Progress in economics often amounts to unify contradicting theories in a more general one with less unobservables $\boldsymbol{s}_{M_i} \cup \boldsymbol{s}_{M_j} \supset \boldsymbol{s}_{M_k}$. Bayesian statistics unifies the parameter estimation and model selection and incorporates the parameter and model uncertainty in a coherent decision framework.
Classical estimates are based on the likelihood $\mathcal{L}(\boldsymbol{\theta}_{M_i}; \boldsymbol{y}) \equiv p(\boldsymbol{y} \,|\, \boldsymbol{\theta}_{M_i}, M_i)$. Parameter estimates are obtained from the likelihood evaluated at the observed sample $\mathcal{L}^0(\boldsymbol{\theta}_{M_i}) \equiv \mathcal{L}(\boldsymbol{\theta}_{M_i}; \boldsymbol{y}^0)$ for example as the parameter vector maximizing the likelihood $\hat{\boldsymbol{\theta}}_{M_i} = \text{argmax}\ \mathcal{L}^0(\boldsymbol{\theta}_M)$. Often ad hoc criteria are used to obtain estimators for example when minimizing the distance between theoretical and empirical impulse response functions $\hat{\boldsymbol{\theta}}_{M_i} = \text{argmin}\ d(\boldsymbol{\theta}_{M_i}, \boldsymbol{y}^0)$. Classical statistics bases inference on the likelihood, a density conditional on unobservables, by comparing it to data. Bayesian statistics is based on a density conditioned on observables and thereby incorporate the uncertainty about unobservables in the decision. A complete Bayesian model M specification with the variables of interest, density of observables and prior density provides the joint density

$$p(\boldsymbol{\omega}, \boldsymbol{y}, \boldsymbol{\theta}_M \,|\, M) = p(\boldsymbol{\omega} \,|\, \boldsymbol{y}, \boldsymbol{\theta}_M, M)\, p(\boldsymbol{y} \,|\, \boldsymbol{\theta}_M, M)\, p(\boldsymbol{\theta}_M \,|\, M).$$

The posterior of unobservables originates from its prior transformed by the likelihood and the marginal likelihood according to the Bayes formula

$$p(\boldsymbol{\theta}_{M_i} \,|\, \boldsymbol{y}^0, M_i) = \frac{p(\boldsymbol{y} \,|\, \boldsymbol{\theta}_{M_i}, M_i)\, p(\boldsymbol{\theta}_{M_i} \,|\, M_i)}{p(\boldsymbol{y}^0 \,|\, M_i)}.$$

[2] Bold symbols are vector or matrix valued. $p(y|x)$ is a conditional density and may also represent a deterministic function $y = f(x)$.

Evidence of a model updates information about unobservables normalized by the marginal likelihood. The marginal likelihood of a model is given by

$$p\,(\,\boldsymbol{y}^0\,|\,M_i\,) = \int p\,(\,\boldsymbol{y}^0\,|\,\boldsymbol{\theta}_{M_i}, M_i\,)\,p\,(\,\boldsymbol{\theta}_{M_i}\,|\,M_i)\,d\boldsymbol{\theta}_{M_i}\,.$$

It allows data to assign probabilities to models $M = \{M_1, ..., M_m\}$

$$p\,(\,M_i\,|\,\boldsymbol{y}^0,\,M\,) = \frac{p\,(\,\boldsymbol{y}^0\,|\,M_i\,)\,p\,(\,M_i\,)}{p\,(\,\boldsymbol{y}^0\,|\,M\,)} = \frac{p\,(\,\boldsymbol{y}^0\,|\,M_i\,)\,p\,(\,M_i\,)}{\sum_{j=1}^{m} p\,(\,\boldsymbol{y}^0\,|\,M_j\,)\,p\,(\,M_j\,)}.$$

The ratio of two marginal likelihoods is the Bayes factor. It transforms the prior information about the estimated models into the posterior odds ratio

$$\frac{p\,(\,M_i\,|\,\boldsymbol{y}^0\,)}{p\,(\,M_j\,|\,\boldsymbol{y}^0\,)} = \frac{p\,(\,M_i\,)}{p\,(\,M_j\,)}\frac{p\,(\,\boldsymbol{y}^0\,|\,M_i\,)}{p\,(\,\boldsymbol{y}^0\,|\,M_j\,)}.$$

The variable of interest, weighted by the posterior density of unobservables

$$p\,(\,\boldsymbol{\omega}\,|\,\boldsymbol{y}^0,\,M_i\,) = \int p\,(\,\boldsymbol{\omega}\,|\,\boldsymbol{y}^0,\,\boldsymbol{\theta}_{M_i},\,M_i\,)\,p\,(\,\boldsymbol{\theta}_{M_i}\,|\,\boldsymbol{y}^0,\,M_i\,)\,d\boldsymbol{\theta}_{M_i}$$

and the model density

$$p\,(\,\boldsymbol{\omega}\,|\,\boldsymbol{y}^0,\,M\,) = \sum_{i=1}^{m} p\,(\,\boldsymbol{\omega}\,|\,\boldsymbol{y}^0,\,M_i\,)\,p\,(\,M_i\,|\,\boldsymbol{y}^0,\,M\,),$$

implies the expected utility

$$\mathrm{E}\,(\,U\,(\,\boldsymbol{\omega},\,\boldsymbol{x}\,)\,|\,\boldsymbol{y}^0,\,M) = \int U\,(\,\boldsymbol{\omega},\,\boldsymbol{x}\,)\,p\,(\,\boldsymbol{\omega}\,|\,\boldsymbol{y}^0,\,M\,)\,d\,\boldsymbol{\omega}.$$

In economics the basic decision unit is a household with prices and consumption as a variable of interest. In a decision with a researcher and institutions as decision units $\mathcal{D} = \{\,\mathcal{D}^R,\,\mathcal{D}^I\,\}$ the involved priors and models can in general be different. Muth (1961) pointed towards the need to base the explanation of the behavior of institutions on the same model as the researcher and used for this purpose rational expectations. By that he overshot the mark since researchers only search for the appropriate model without knowing it for sure. The assumption of different models for the researcher and the observed opens the door to a robust control analysis and provides the technical means to complete the underlying idea of rational expectations by taking into account the differences in the informational endowment of the researcher and the observed.
The variables of interest in economics are causal relationships between observables and unobservables. Models, parameter estimates and specification test statistics like model selection criteria are the available policies. Econometrics is based on

conditional independence assumptions. Their implications for causality are still insufficiently understood and therefore not formalized. The reason lies certainly in enduring problems of a satisfactory definition of causality in philosophy. The symptom is that the most sophisticated instruments available for this matter in econometrics are exogeneity and Granger causality. Artificial intelligence research depends on a detailed formalization for causality inference in order to transform computers into more than fast calculating silicon. It goes well beyond exogeneity and Granger causality analysis. The Bayesian information accounting allows to encode the conditional independence assumptions in Bayes networks and infer causal relationships as a game against nature. The graph theory and further formalisms allow to analyze the networks for causality inferences. Many concepts in econometrics like Granger causality, instrument variables, identification, Lucas' critique, Occam's razor, spurious regressions or counterfactual policies are related in a formal framework to derive probabilities of causation. It transcend the usual hand waving treatment of these difficult concepts. Pearl (2000) introduces to this most fascinating reasoning which might shape the future developments in econometrics towards a more conscious causality inference. Together with the statistical decision framework of a Bayesian approach the production function of econometrics and the economics of economics is advancing.

The eight formulas summarizing the statistical decision theory are only apparently simple. The computational challenges of a general nonlinear Bayesian likelihood approach are substantial. Many high dimensional implicit functions and integrals have to be approximated to obtain one single likelihood evaluation at some structural parameters. A practical computational problem is the curse of dimensionality common to most numerical algorithms. It implies that the computational effort rises exponentially with the dimension of the problem. This is very relevant for example in international macroeconomics where theoretical models go well beyond the standard model with two countries each represented by one agent and one technology or portfolio decisions where many asset prices have to be modelled.

The curse of dimensionality appears if a univariate approximation and integration operator is extended from the univariate to the multivariate case by the Kronecker product. A continuous random shock can be approximated for example by a discrete variable with five possible realizations. For a model with 30 shocks, $5^{30} > 9 \times 10^{20}$ integrand evaluations are needed to approximate the rational expectation for one variable. This huge number of function evaluations is also needed for the approximation of a policy of 30 variables by a forth order polynomial. In order to solve the model it has to be done some 100 times to converge within a root finding algorithm at an implicitly defined policy function. For a given vector of structural parameters the policy function implies a likelihood. It is the sum of period contributions and each is a multivariate nonlinear integral. A maximum likelihood estimation will have to evaluate the likelihood at some 100 parameter vectors in addition to many evaluations needed to obtain numerical

first derivatives. For a Bayesian estimate of the posterior density of structural parameters some 10,000 likelihood evaluations are needed within a simulation based density approximation. Computational speed considerations are obviously an important topic and gains allow to go beyond highly stylized models.
An intuition of the Smolyak operator can be gained by looking at a multidimensional Taylor expansion. It does not use Kronecker products of univariate polynomials to obtain multivariate polynomials. Instead they are constructed as products of univariate polynomials in such a way that the sum of the exponents of the univariate factors is beneath a certain number which characterizes the approximation accuracy. These polynomials are already known as complete polynomials, see for example in Judd (1998), but it was not clear how to obtain the points where the approximated function has to be evaluated. Since complete polynomials are a special case of the Smolyak operator it can be used to operate on the points for function evaluation in one dimension in order to construct the points in many dimensions.
Another interesting feature of the operator is its hierarchical structure which offers an estimator of the approximation accuracy so that an accuracy depending stopping criterion can be applied instead of an à priori given degree of approximation. This feature can be implemented easily for the integration operator by adding more evaluation points and reweighting the integrand at the old points. It is potentially useful in the estimation process where the model is solved at many different parameter vectors. Some of them may require only a low degree of approximation for a sufficient accuracy while others may imply a higher degree.
The general model representation is the one used by Miranda and Fackler (2002)

$$\begin{aligned} \mathbf{0} &= \boldsymbol{f}(\boldsymbol{s}_t, \boldsymbol{x}_t, E_t\boldsymbol{h}(\boldsymbol{s}_t, \boldsymbol{x}_t, \boldsymbol{e}_{t+1}, \boldsymbol{s}_{t+1}, \boldsymbol{x}_{t+1})) \\ \boldsymbol{s}_{t+1} &= \boldsymbol{g}(\boldsymbol{s}_t, \boldsymbol{x}_t, \boldsymbol{e}_{t+1}). \end{aligned}$$

The variables have to be grouped into state $\boldsymbol{s}$ and policy $\boldsymbol{x}$ vectors. The structural state shocks $\boldsymbol{e}$ are in general not Gaussian and independently distributed. $\boldsymbol{f}$ are dynamic first order optimality conditions and $\boldsymbol{g}$ are specified state transition functions. Economic models are functional equations and solutions are policy functions in terms of states $\boldsymbol{x}_t = \boldsymbol{x}^*(\boldsymbol{s}_t)$. The policy functions are defined only implicitly in the first order conditions and can usually not be derived in closed form. Therefore numerical approximation methods have to be used.
The general approach to solve functional equations is to restrict the search for the solution function in the infinite dimensional space of all functions to a lower dimensional space. Functions within this smaller space are represented by a vector of parameters. They are calculated to fit a linear combination of simple basis functions to policy function values at a certain grid. To identify these parameters the function which is being approximated needs to be evaluated at some points. This is a similar problem as in basic econometrics, where these points and the function evaluations are given by data for the independent and dependent variables, the functional form is linear and the criterion to obtain the

parameters is least squares. In numerical functional problems we can choose the points where to evaluate the function, the functional form fitted to data and the criterion to identify the parameters of the approximation.
In economic models the policy functions are defined only implicitly and the evaluation of the policy functions cannot be done directly. For a given vector of structural parameters the first order conditions are evaluated at a grid of states and policies. The policy values are changed by function iteration or some general nonlinear root finding algorithm until the first order conditions are near the theoretically exact value of zero.
In an estimation algorithm tens of thousands of nonlinear solutions have to be calculated. Beside speed considerations there are two important topics within this process. One is about finding sensible start values for the policy functions and the other dwells on a robust way to arrive at a solution. In calibration applications where the model is solved for some few parameter vectors this can be handled by searching manually for start values which converge to a plausible solution checked by suitable approximation error estimates. This manual procedure is not practicable in econometric applications and a solver relying on numerical linearization is programmed to generate start values for the nonlinear solver. The likelihood of the linearized model is moreover evaluated by the Kalman filter and provides a check whether an alternative nonlinear solution and estimation improves the fit. The relative fit of these models is examined by the marginal likelihood which is suited as a model selection criterion for nonnested models.
After the solution of a model is obtained, its reduced form can be given in the state space form in order to evaluate the likelihood. It is a general time series model and its most attractive features are the distinction between observed and unobserved or latent variables and a recursive updating of information based on the Bayes' formula and the conditional independence property of the involved Markovian process. This allows an integrated approach to nonstationary and cointegrated processes, stochastic trends like the one assumed by the Hodrick-Prescott filter, missing observations, regime switching, GARCH process or in general time varying parameters, learning processes, data revision and robust control applications. In the general nonlinear state space model the distinction between a parameter and a state becomes blurred. Both are unobservables and a parameter can be assumed to be time varying described by some process. By that way the original fixed parameter becomes an unobserved state and the process describing the time dependency is itself described by some fixed parameters.
The solution of the model $\boldsymbol{x}_t = \boldsymbol{x}^*(\boldsymbol{s}_t)$ implies dynamics given by the state transition equation

$$\boldsymbol{s}_{t+1} = \boldsymbol{g}(\boldsymbol{s}_t, \boldsymbol{x}^*(\boldsymbol{s}_t), \boldsymbol{e}_{t+1}) = \boldsymbol{g}^*(\boldsymbol{s}_t, \boldsymbol{e}_{t+1})$$

which together with a measurement equation

$$\boldsymbol{y}_t = \boldsymbol{m}(\boldsymbol{s}_t, \boldsymbol{\epsilon}_t)$$

forms the state space representation of the model. The measurement equation links the unobservable states to observables $\boldsymbol{y}$ with measurement errors $\boldsymbol{\epsilon}$.
The likelihood of a state space model can be obtained as a by-product when deriving the posterior densities of the unobserved states recursively for each period. The densities involved in updating are classified according to the conditioning set. $p\left(\boldsymbol{s}_\tau \mid \boldsymbol{y}_{1:t}\right)$ is the prior or prediction density for $\tau > t$, filter or posterior density for $\tau = t$ and smoothing density for $\tau < t$. The prior is updated to become the posterior after new data is incorporated and the smoothing density is obtained after the state densities are conditioned on all available data.
The posteriors can be estimated by a bootstrap filter which allows for any shock distribution and nonlinear functions $\boldsymbol{g}$ and $\boldsymbol{m}$. The bootstrap filter approximates the posterior by simulating complete densities. It is a good accuracy benchmark but computationally involved due to its general nature. The likelihood is derived by multivariate integral equations and the particle filter is a Monte Carlo integration method. It has to be combined with a random number generator since the difficulty is to derive draws from a density without a closed form expression. If functions $\boldsymbol{f}, \boldsymbol{g}, \boldsymbol{m}$ and $\boldsymbol{x}^*$ are smooth, the computational efficiency can be gained by exploiting this property.
An overview of some available approaches to nonlinear filtering is given. The general filtering problem is formulated and alternative solutions are discussed. The purpose of this part is to work out where integral equations arise and deterministic integration by the Smolyak operator is possible. The most common deterministic approach is the unscented filter developed by Julier and Uhlmann (1997) and refined in Julier and Uhlmann (2002). It uses a low order approximation to the integrals involved in order to calculate two moments of a nonlinear transformation of a Gaussian random variable. Structural econometric models result in state transition functions $\boldsymbol{g}$ where the policy solution is plugged in. These policy functions are often of a higher polynomial degree than the unscented filter is derived for. Filtering needs to integrate the state transition functions and the unscented integration is probably not accurate enough for structural econometric applications.
The unscented Kalman and the bootstrap filter can be classified as two possible extreme approaches. The former uses only two moments of a low order approximation for filtering whereas the latter uses complete simulated densities. In general a filter with a flexible approximation degree is desirable since models can exhibit different degrees of nonlinearity and smoothness. A quadrature based nonlinear filter with different degrees of approximation is therefore a compromise between these two extremes and thereby allows to trade off the model dimension against smoothness.
Once a likelihood evaluation procedure is established it can be used for a parameter estimation. In this work I estimate the posterior density of structural parameters by a Metropolis-Hastings algorithm. The parameter posterior is proportional to the likelihood times prior. A posterior density without a closed

form can be approximated by a histogram of a sample of random draws. The Metropolis-Hastings algorithm offers a general method to draw random numbers from a density. The only prerequisite is that the density can be evaluated at any point in its domain. The algorithm travels through the feasible parameter space and a simple criterium decides whether the next candidate vector is accepted as a new realization from the posterior or whether the last vector is used again in order to generate another candidate.
Fernádez-Villaverde and Rubio-Ramírez (2004b) use a Metropolis-Hastings algorithm with one sequence of structural parameters generated by a random walk through the parameter space. It is a Markov Chain because only the last parameter vector determines how the next proposal vector is generated. The variances of the random walk shocks have to be tuned before estimation. The recommendation is to choose them in such a way that the sequence exhibits the optimal acceptance ratio around 0.3. The acceptance ratio measures how many candidate vectors are accepted as a fraction of all draws. A higher variance lowers the ratio and vice versa. It is quite difficult to tune these variances in a satisfactory way for more than one parameter.
Compared to a likelihood approach within a derivative based maximization routine there are several advantages of the Metropolis-Hastings algorithm. The derivative based maximization may not converge and end up in local maxima. Either different start values have to be tried manually or a global approach like homotopy or genetic methods have to be implemented. The Metropolis-Hastings algorithm is easy to implement. It is also a simple global genetic hill climber. Another advantage is that estimated uncertainty about the parameters does not rely on the asymptotic theory and sample size dependent uncertainty estimates are obtained. The uncertainty about parameter estimates is moreover properly incorporated in the decision. as opposed to classical econometrics where some bootstrap approaches have to be used. This is important for example in forecasts or for calibrated parameters. The common calibration approaches to empirical model evaluation can also be improved by incorporating the parameter uncertainty in the prior density. Finally the parameter posterior sample generated with this algorithm provides the starting point for the density estimate of any variables of interest. It can be any function of observables and unobservables like specification test statistics, welfare measures for the policy recommendation and evaluation.
I present a genetic extension of the basic random walk Metropolis-Hastings algorithm to solve its covariance choice problem in an automatic fashion. The innovation is to combine several parallel sequences and to allow each sequence to be driven by random walk innovations and a mixture of parameter vectors from the parallel sequences. This genetic extension reduces the number of free parameters to be tuned to two, so that the recommended acceptance ratio is obtained with less and less expensive trial sequences. The intuition of the algorithm extension can be related to the way a standard diagnostic test for convergence

of the algorithm is obtained. Between and within moments should converge to assure that several sequences are drawn from the same density. Since the optimal random walk shock variance is the variance of the unknown target density one might think that the genetic extension allows to estimate the appropriate covariance matrix from parallel sequences. The parallel estimate cannot be obtained from one sequence since parallel sequences start from different parameter vectors. The global maximization properties of a one sequence algorithm are improved and a global likelihood maximizer can be used as an alternative to the posterior density estimation. I used it as as the first step to find the modes of the estimated density before the sampling starts. Moreover parallel sequence allow for an unbiased convergence test. The diagnostic test is needed to assure that the algorithm has converged and the generated parameter vectors represent draws from the posterior density independent of start values.
Two other properties within the Metropolis-Hastings algorithm are exploited to accelerate the estimation. Once the sequences have converged subsequent structural parameter vector draws differ only by small amounts and the policy function at the previous vector can be used as a start value. This feature combined with a derivative based root finding algorithm reduces the computations for solving the model to only a small fraction of the overall computations. The main work is the likelihood evaluation for a given structural parameter vector and its associated policy function.
The second acceleration results from the use of constant approximation bounds for the solution algorithm as well as the estimation process. This implies in a constant inverse matrix for calculating the Chebyshev coefficients. Fast Fourier transformation and Smolyak specific implementations decrease only the fixed costs of the matrix inversion.
The possibility of the Bayesian framework to select models regarding their ability to fit the data also seems interesting enough. The standard procedure is an informal check how the model fits some stylized facts, a likelihood ratio test or a information criterion like BIC. The drawback of a likelihood ratio test is that the models have to be nested in such a way that one emerges from the other by a parameter restriction. Moreover these tests rely on the fiction of a true model and their small sample properties are not clear either. The Bayesian information criterion is in fact an approximation to the Bayes factor used here and the posterior odds ratio is the proper decision theoretical model selection criterion. As in Fernández-Villaverde and Rubio-Ramírez (2004a) it can be estimated once the Metropolis-Hastings algorithm has converged and produced a sample from the posterior density of structural parameters. With the construct of the variable of interest at hand, out-of-sample forecast performance of a model can also be measured. Such a forecast for future periods $\boldsymbol{y}_t$ with $t > T$ is just another unobservable to be simulated by means of the posterior density.

3 Optimal Policy

The general model representation in this work is the one used in Miranda and Fackler (2002). It is summarized in table 1. The vector valued function $\boldsymbol{f}$ repre-

Table 1: General Macroeconomic Model

Model			
$\boldsymbol{f}(\boldsymbol{s},\boldsymbol{x},\boldsymbol{z})$	$=$	$\boldsymbol{0}$	
$\boldsymbol{z}$	$=$	$\mathrm{E}_{e'}\boldsymbol{h}(\boldsymbol{s},\boldsymbol{x},\boldsymbol{e}',\boldsymbol{s}',\boldsymbol{x}')$	
$\boldsymbol{s}'$	$=$	$\boldsymbol{g}(\boldsymbol{s},\boldsymbol{x},\boldsymbol{e}')$	
Variables			
$\boldsymbol{s} \in S$	$\subseteq$	$\mathbb{R}^{d_s}$	state variables
$\boldsymbol{x} \in [\boldsymbol{a}(\boldsymbol{s}),\boldsymbol{b}(\boldsymbol{s})]$	$\subseteq$	$\mathbb{R}^{d_x}$	policy variables
$\boldsymbol{z} \in$		$\mathbb{R}^{d_z}$	expectational variables
$\boldsymbol{e} \in$		$\mathbb{R}^{d_e}$	stochastic shocks
Model Functions			
$\boldsymbol{f} : \mathbb{R}^{d_s+d_x+d_z}$	$\to$	$\mathbb{R}^{d_x}$	equilibrium conditions
$\boldsymbol{h} : \mathbb{R}^{d_s+d_x+d_e+d_s+d_x}$	$\to$	$\mathbb{R}^{d_z}$	expectation functions
$\boldsymbol{g} : \mathbb{R}^{d_s+d_x+d_e}$	$\to$	$\mathbb{R}^{d_s}$	state transitions
$\boldsymbol{x}^e : \mathbb{R}^{d_s+d_z}$	$\to$	$\mathbb{R}^{d_x}$	expectational solution
Approximated Functions			
$\boldsymbol{f}(\boldsymbol{s},\boldsymbol{x},\mathrm{E}_{e'}\boldsymbol{h}(\boldsymbol{s},\boldsymbol{x},\boldsymbol{e}',\boldsymbol{g}(\boldsymbol{s},\boldsymbol{x},\boldsymbol{e}'),\boldsymbol{x}')) = 0$			
$\boldsymbol{x}^* : \mathbb{R}^{d_s}$	$\to$	$\mathbb{R}^{d_x}$	policy functions
$\boldsymbol{f}(\boldsymbol{s},\boldsymbol{x}^*(\boldsymbol{s}),\sum_j w_j\boldsymbol{h}(\boldsymbol{s},\boldsymbol{x}^*(\boldsymbol{s}),e'_j,\boldsymbol{s}'_j,\boldsymbol{x}^*(\boldsymbol{s}'_j)) = 0$			
$\boldsymbol{s}'_j = \boldsymbol{g}(\boldsymbol{s},\boldsymbol{x}^*(\boldsymbol{s}),\boldsymbol{e}'_j)$			
$\boldsymbol{z}^* : \mathbb{R}^{d_s}$	$\to$	$\mathbb{R}^{d_z}$	expectation functions
$\boldsymbol{f}(\boldsymbol{s},\boldsymbol{x}^e(\boldsymbol{s},\boldsymbol{z}^*(\boldsymbol{s})),\sum_j w_j\boldsymbol{h}(\boldsymbol{s},\boldsymbol{x}^e(\boldsymbol{s},\boldsymbol{z}^*(\boldsymbol{s})),\boldsymbol{e}'_j,\boldsymbol{s}'_j,\boldsymbol{x}^e(\boldsymbol{s}'_j,\boldsymbol{z}^*(\boldsymbol{s}'_j))) = \boldsymbol{0}$			
$\boldsymbol{s}'_j = \boldsymbol{g}(\boldsymbol{s},\boldsymbol{x}^e(\boldsymbol{s},\boldsymbol{z}^*(\boldsymbol{s})),\boldsymbol{e}'_j)$			
time is denoted by $v \equiv v_t$, $v' \equiv v_{t+1}$ for $v = \boldsymbol{s},\boldsymbol{x},\boldsymbol{z},\boldsymbol{e}$			

sents the necessary first order conditions of a dynamic optimization. Distorted, decentralized and social planner equilibrium models can be written in this form and encompass the majority of currently used macroeconomic models. The first order conditions for a social planner equilibrium can be derived by the Bellman functional. The vector valued function $\boldsymbol{g}$ represents the state transition laws of the economy which are shock processes and other state transitions, like the productivity and capital transition in the example model. $\boldsymbol{h}$ represents the forward looking part of the model. The variables have to be classified as state $\boldsymbol{s}$ or policy $\boldsymbol{x}$ (also called action, response or decision variable). The difference is that the transition laws for the state variables are given as part of the model specification

whereas the optimal reaction of the policy variables with respect to the states are the object of interest when solving the model.
Mathematically the equation is a functional, like integral or differential equations, and the solution is the policy function. It prescribes how policy should be conducted depending on the state variables so that it satisfies the first order conditions and is therefore dynamically optimal. Since these conditions are usually nonlinear and contain an expectation operator they cannot be explicitly solved for the policy and approximation methods are to be used.
There exist at least two possible approximation strategies within the general model. The first approximates the policy functions $\boldsymbol{x}^*(\boldsymbol{s})$ and the second the expectations $\boldsymbol{z}^*(\boldsymbol{s})$. Both approaches are equivalent since $\boldsymbol{x}^*(\boldsymbol{s})$ can be recovered from $\boldsymbol{h}(\boldsymbol{s}, \boldsymbol{x}(\boldsymbol{s}))$ and vice versa. I implemented the policy function approximation. The notion $\boldsymbol{x} \in [\boldsymbol{a}(\boldsymbol{s}), \boldsymbol{b}(\boldsymbol{s})]$ allows for state dependent inequality constraints in the model formulation. The resulting Kuhn-Tucker inequality conditions can be transformed into nonlinear max/min equations with smoothed kinks in such a way that they can be solved by a usual root finder. Inequality restrictions are for example liquidity constraints in OLG models saying that young generations cannot borrow against future income prospects. This will result in kinks in the policy functions. Such functions cannot be approximated very well with global polynomials and finite elements should be used. Both are discussed in section 3.3 but in the current software only global polynomials are implemented. However, a finite element approximation is a rather simple extension of the current implementation.

3.1 Utility Maximization

The example model is the canonical dynamic consumption and labor choice decision. It is described by the following constrained maximization

$$\max_{\{c_t, l_t\}_{t=0}^{\infty}} U = \mathrm{E}_0 \sum_{t=0}^{\infty} \beta^t u(c_t, l_t) = \mathrm{E}_0 \sum_{t=0}^{\infty} \beta^t \frac{(c_t^\theta (1-l_t)^{1-\theta})^{1-\tau}}{1-\tau} \tag{1}$$

subject to

$$y_t = c_t + i_t \tag{2}$$

$$y_t = e^{a_t} k_t^\alpha l_t^{1-\alpha} \tag{3}$$

$$k_{t+1} = i_t + (1-\delta) k_t \tag{4}$$

$$a_{t+1} = \rho a_t + e_{t+1} \quad \text{where } e_t \overset{iid}{\sim} \mathcal{N}(0, \sigma_e). \tag{5}$$

The variable of interest in the decision of equation (1) is the maximum of the present value of the utility from consumption c_t and leisure $1 - l_t$ policy. Leisure is given by the time remaining after labor l_t is subtracted from the available time normed to 1. The structural parameters are the discount factor β, the elasticity of intertemporal substitution τ, the labor supply determinant θ, the technical

substitution α, the depreciation rate δ and the autocorrelation coefficient ρ of the productivity process a_t. E_0 are expectations conditional on available information at time 0. Equation (2) is the budget constraint, restricting output y_t to be either consumed c_t or invested i_t. Equation (3) describes the production technology with capital k_t, labor l_t and productivity a_t. Equation (4) is the law of motion of capital and equation (5) describes the process of productivity with productivity shock e_t.

3.2 First Order Conditions

The model has no distortions. Hence both welfare theorems apply and the necessary first order conditions can be derived by the Bellman functional as an optimization problem of a social planner. To facilitate the derivation, I substitute two out of three constraints by plugging (3) and (4) into (2) and eliminate y_t and i_t. The Bellman equation specifies the value function

$$\mathrm{V}(k_t, a_t) = \max_{c_t, l_t, k_{t+1}} u(c_t, l_t) + \beta\, \mathrm{E}_t \mathrm{V}(k_{t+1}, a_{t+1}) + \lambda(k_{t+1} - e^{a_t} k_t^{\alpha} l_t^{1-\alpha} + c_t - (1-\delta)k_t).$$

The first derivatives with respect to the decision variables are

$$\frac{\partial \mathrm{V}(k_t, a_t)}{\partial c_t} = \theta \frac{(c_t^{\theta}(1-l_t)^{1-\theta})^{1-\tau}}{c_t} + \lambda \stackrel{!}{=} 0 \tag{6}$$

$$\frac{\partial \mathrm{V}(k_t, a_t)}{\partial l_t} = -(1-\theta)\frac{(c_t^{\theta}(1-l_t)^{1-\theta})^{1-\tau}}{1-l_t} - \lambda(1-\alpha)\frac{e^{a_t} k_t^{\alpha} l_t^{1-\alpha}}{l_t} \stackrel{!}{=} 0 \tag{7}$$

$$\frac{\partial \mathrm{V}(k_t, a_t)}{\partial k_{t+1}} = \beta \mathrm{E}_t \mathrm{V}_{k_{t+1}}(k_{t+1}, a_{t+1}) + \lambda \stackrel{!}{=} 0. \tag{8}$$

The derivative $\mathrm{V}_{k_{t+1}}(k_{t+1}, a_{t+1})$ has to be substituted since the value function is unknown. The envelope theorem allows to omit the derivatives of the unknown policy function with respect to the states since at optimum they are zero. The derivative of the value function is therefore

$$\begin{aligned} \mathrm{V}_{k_t}(k_t, a_t) &= -\lambda(1 - \delta + \alpha \frac{e^{a_t} k_t^{\alpha} l_t^{1-\alpha}}{k_t}) \\ &= \theta \frac{(c_t^{\theta}(1-l_t)^{1-\theta})^{1-\tau}}{c_t} \left(1 - \delta + \alpha \frac{e^{a_t} k_t^{\alpha} l_t^{1-\alpha}}{k_t}\right) \end{aligned}$$

where λ is substituted by equation (6). The derivative can be forwarded one period and used to equate (8) and (6) through λ. This gives the first out of two necessary first order condition. It is an Euler equation which characterizes the optimal intertemporal consumption and labor policy

$$\frac{(c_t^{\theta}(1-l_t)^{1-\theta})^{1-\tau}}{c_t} - \beta \mathrm{E}_t \left(\frac{(c_{t+1}^{\theta}(1-l_{t+1})^{1-\theta})^{1-\tau}}{c_{t+1}} \left(1 - \delta + \alpha \frac{e^{a_{t+1}} k_{t+1}^{\alpha} l_{t+1}^{1-\alpha}}{k_{t+1}}\right)\right) = 0.$$

(9)

An optimal decision balances present and future marginal rewards. The present reward is simply the present marginal utility whereas the future reward is the future marginal utility times the gross return which depends on the discount factor, depreciation and the marginal productivity of capital. Using (6) and (7) to eliminate λ gives us the second necessary first order condition

$$\frac{1-\theta}{\theta}\frac{c_t}{1-l_t} - (1-\alpha)\frac{e^{a_t}k_t^{\alpha}l_t^{1-\alpha}}{l_t} = 0. \tag{10}$$

It is a static equation defining an optimal intratemporal trade off between consumption and labor.
The numerical procedures can be used to find two policy functions for labor and consumption implicitly defined in the last two equations. This is inefficient since the static optimality condition can be solved explicitly for consumption in terms of labor. The solution can be plugged into the Euler equation turning it into an equation describing the optimal labor supply. Once the optimal labor decision is approximated the optimal consumption can be recovered from the static equation. In order to set out the problem in a general multivariate setting I do not use this analytical solution in the discussion but only in the numerical routines.
Now the system can be mapped into the general form of table 1. The state vector is $\boldsymbol{s}_t \equiv \{k_t,\, a_t\}$ with $d_s = 2$, the policy variables are $\boldsymbol{x}_t \equiv \{c_t, l_t\}$ with $d_x = 2$ and the productivity shock in equation (5) corresponds to the shock in the general model form, $\boldsymbol{e}_t \equiv e_t$ with $d_e = 1$. There is only one expected variable $\boldsymbol{z}_t$ with $d_z = 1$ and its functional form $\boldsymbol{h}$ is given by the argument of the expectation operator in equation (9). Equations (9) and (10) correspond to the vector valued function $\boldsymbol{f}$ with $d_x = 2$. The capital and productivity transition equations

$$\begin{aligned} k_{t+1} &= e^{a_t}k_t^{\alpha}l_t^{1-\alpha} - c_t + (1-\delta)k_t \\ a_{t+1} &= \rho a_t + \epsilon_t \end{aligned}$$

correspond to the vector valued state transition function $\boldsymbol{g}$ with $d_s = 2$. The model solution $\boldsymbol{x}^*(\boldsymbol{s})$ implies the state transition

$$\boldsymbol{s}_{t+1} = \boldsymbol{g}(\boldsymbol{s}_t, \boldsymbol{x}^*(\boldsymbol{s}_t), \boldsymbol{e}_t) = \boldsymbol{g}^*(\boldsymbol{s}_t, \boldsymbol{e}_t).$$

This is the state transition equation of a state space model and will be discussed in section 4 as a tool to evaluate the likelihood of the model.

3.3 Approximation

The solution of a functional is a function and resides in an infinite dimensional space. A numerical procedure has to restrict its search to a lower dimensional space where the solution is characterized by a finite vector of parameters $\boldsymbol{c}$. The

univariate function approximation is given by a linear combination of basis functions $\psi_i(s)$ of some functional form

$$f(s) \approx \hat{f}(s) = \sum_{i=0}^{n} c_i \, \psi_i(s). \tag{11}$$

However, the policy functions $\boldsymbol{x}^*(\boldsymbol{s})$ depend in general on d_s state variables and some further techniques are needed to extend approximation to the multidimensional case. There are three broad numerical methods for function approximation. The perturbation approach is multidimensional by nature. The finite element and spectral approaches are univariate in the beginning and have to be extended to a multivariate operator. The extension methods compared in this work are the Kronecker product and the Smolyak operator and will be discussed in section 3.5. The most popular perturbation method uses Taylor series expansions of the model equations to derive the policy function. Another perturbation method uses Padé series which are ratios of Taylor series. Often a loglinear approximation is used. Whether this is an improvement can be answered by approximation error estimates and Aruoba, Fernádez-Villaverde, and Rubio-Ramírez (2003) report that for the model used here loglinearization exhibits larger approximation errors. Judd (2002) generalizes this change of variable and shows that the optimal transformations are difficult to derive but dominant can be constructed. The predominant linearization technique is a perturbation method where a first order Taylor expansion around the deterministic steady state is calculated. Perturbation methods rely on the implicit function theorem and use only local information, namely the policy and its first and higher derivatives at the steady state.
One problem is that the approximation is valid only within a ball around the steady state and a radius equal to the distance between the steady state and the next singularity of the function approximated. Valid means that outside this ball the approximation quality is unacceptable. Since the functions we are approximating are defined only implicitly, the singularity can be hardly obtained analytically and an approximation error estimate is indispensable. This is valid for all approximation strategies and one possible error estimate is discussed in section 3.9.
Taylor series can be used to gain economic intuition about the dynamic mechanism at work since the functional form in the implied dynamic system $\boldsymbol{s}_{t+1} = \boldsymbol{g}(\boldsymbol{s}_t, \boldsymbol{x}^*(\boldsymbol{s}_t), \boldsymbol{e}_t)$ is known and the parameters are explicitly expressed as functions of the structural parameters. A second order Taylor approximation can be used to avoid imposing the certainty equivalence property on a nonlinear model. Then the policy depends on the state vector and the state shock standard deviations. The perturbation approach is described by Schmitt-Grohé and Uribe (2004) and is now routinely used since software for a general class of models is available. A disadvantage is that higher order approximations are said to require analytical derivatives because numerical derivatives accumulate errors. They should

be calculated by symbolic software like Maple or Mathematica. I calculate the linearization by numerical derivatives. The error estimates show that analytical derivatives do hardly improve the approximation quality at the true parameters and do not change the likelihood at all. Another disadvantage is that kinks of the policy function and therefore inequality restrictions cannot be handled.
For the spectral and finite element approximation the borders of the approximation space $S \ni \boldsymbol{s}$ have to be defined. It is not necessary to calculate the steady state for the approximation but it gives an idea about the appropriate approximation interval $S \subset \mathbb{R}^{d_s}$.
The spectral approach use orthogonal polynomials instead of nonorthogonal monomials ($\psi_n(x) = x^n$) of the Taylor expansion. A family of polynomials $\{\psi_n(x)\}$ is mutually orthogonal with respect to a weighting function $w(x)$ if

$$< \psi_n(x), \psi_m(x) > \quad = \int_a^b \psi_n(x)\, \psi_m(x)\, w(x)\, dx = 0 \text{ for } n \neq m.$$

The coefficients of the spectral polynomial are derived from function evaluations at more points in the approximation space then only the steady state as in the perturbation approach. The Chebyshev approximation comes close to the holy grail of the approximation theory: the minimax polynomial. It is the approximating polynomial which has the smallest maximum deviation from the true function. It is closely approximated by Chebyshev polynomials of the first kind used in this work. The disadvantage of this method is that it is not suited to handle kinks in the policy functions. The advantage is that for smooth models it needs less function evaluations compared to the next method. The number of points where the function is evaluated determines the length of the policy defining parameter vector. This vector has to be found within a numerical root finding procedure and shorter vectors results in a much faster algorithm.
The third approximation method is finite elements. It approximates like the spectral method globally but uses local low order polynomials. Finite elements have basis functions which are nonzero only in a small region of the approximation space whereas spectral basis functions are nonzero at almost all points of the approximation space. It effectively divides the approximation space and approximates within these subspaces such that low degree polynomials are sufficiently accurate. The advantage is that it can accurately approximate kinks in policy functions since these local anomalies do not influence the approximation in other subspaces. The disadvantage is that for smooth functions it needs many function evaluations compared to the spectral method.

3.3.1 Perturbation

Perturbation is a local function approximation. In a first order perturbation the policy value and its first derivative are used to recover the implicit policy function. Linear rational expectation models result in a quadratic matrix or

Riccati equation which can be solved fast and accurate by the generalized Schur or QZ decomposition, see Klein (2000). Why do we arrive at a quadratic equation for the parameters which determines the model solution $\boldsymbol{x}^*(s)$? If the first order conditions $\boldsymbol{f}$, the state transitions $\boldsymbol{g}$ and the expectations function $\boldsymbol{h}$ are linear then the solution $\boldsymbol{x}^*(\boldsymbol{s})$ is linear, too. The dynamic nature of the model introduces a nested application of the solution function $\boldsymbol{x'} = \boldsymbol{x}^*(\boldsymbol{s'}) = \boldsymbol{x}^*(\boldsymbol{g}(\boldsymbol{s}, \boldsymbol{x}^*(\boldsymbol{s}), \boldsymbol{e}))$ within the forward looking function $\boldsymbol{h}$. If for example $x^*(s) = a + bs$ than $x^*(x^*(s)) = a + b(a + bs) = a + ab + b^2 s$, i.e. the solution parameter b appears quadratically and the first order functional has two roots. Economists are usually interested in the one which implies a stable saddle path of the underlying variables in $\boldsymbol{s}_{t+1} = \boldsymbol{g}(\boldsymbol{s}_t, \boldsymbol{x}^*(\boldsymbol{s}_t), \boldsymbol{e}_t)$.
A more accurate solution can be obtained if higher order Taylor polynomials and therefore second and higher order derivatives of the policy function at the steady state are used. The policy at the steady state is different compared to the first order approximation since risk premia are taken into account. This is achieved by augmenting the state vector on which the policy depends by the standard deviation of the involved shocks. The marginal effort of a second order approximation is low. Beside another derivative of the system it is a simple matrix inversion. Once the saddle path in the first order Taylor approximation is picked out, the solution for the second order approximation is unique.
Now the linearization equations are given for the general model class. First the deterministic steady state $\bar{\boldsymbol{s}}, \bar{\boldsymbol{x}}, \bar{\boldsymbol{e}}$ is needed. The steady state of shocks is their zero expected value $\bar{\boldsymbol{e}} = \boldsymbol{0}$. For the other variables the deterministic steady state is defined by

$$\begin{aligned} \boldsymbol{0} &= \boldsymbol{f}(\bar{\boldsymbol{s}}, \bar{\boldsymbol{x}}, \boldsymbol{h}(\bar{\boldsymbol{s}}, \bar{\boldsymbol{x}}, \boldsymbol{0}, \bar{\boldsymbol{s}}, \bar{\boldsymbol{x}})) \\ \bar{\boldsymbol{s}} &= \boldsymbol{g}(\bar{\boldsymbol{s}}, \bar{\boldsymbol{x}}, \boldsymbol{0}). \end{aligned}$$

A closed form solution is usually not available and a nonlinear root finder has to solve the equations. The linearized first order equations around the steady state are given by

$$\begin{bmatrix} \boldsymbol{I} & \boldsymbol{0} \\ -\boldsymbol{f}_z \boldsymbol{h}_{s'} & -\boldsymbol{f}_z \boldsymbol{h}_{x'} \end{bmatrix} \begin{bmatrix} d\boldsymbol{s'} \\ d\boldsymbol{x'} \end{bmatrix} = \begin{bmatrix} \boldsymbol{g}_s & \boldsymbol{g}_x \\ \boldsymbol{f}_s + \boldsymbol{f}_z \boldsymbol{h}_s & \boldsymbol{f}_x + \boldsymbol{f}_z \boldsymbol{h}_x \end{bmatrix} \begin{bmatrix} d\boldsymbol{s} \\ d\boldsymbol{x} \end{bmatrix}$$

where $d\boldsymbol{s}$ and $d\boldsymbol{x}$ denote deviations from the steady state and $\boldsymbol{x'}$ and $\boldsymbol{s'}$ are again the next period variables. The dimensions of the identity $\boldsymbol{I}$ and null matrix $\boldsymbol{0}$ are $d_s \times d_s$ and $d_s \times d_x$ respectively. The subscripted functions $\boldsymbol{f}$, $\boldsymbol{g}$ and $\boldsymbol{h}$ are Jacobians with respect to the variables in the subscript evaluated at the steady state. Therefore the left matrices on both sides of the equation are constants for a given vector of structural parameters. The solution is a $d_x \times d_s$ matrix $\boldsymbol{C}$ with $d\boldsymbol{x} = \boldsymbol{C} d\boldsymbol{s}$. This solution implies the state transition matrix $\boldsymbol{P}$ of size $d_s \times d_s$ in $d\boldsymbol{s'} = \boldsymbol{P} d\boldsymbol{s}$. The matrix $\boldsymbol{C}$ represents the optimal linear policy for a given state vector. The matrix $\boldsymbol{P}$ maps the current state into the expectation of next period

state. It is the combined effect of the policy function $\boldsymbol{C}$ in the specified state transition. The resulting matrix system is

$$\begin{bmatrix} \boldsymbol{I} & \boldsymbol{0} \\ -\boldsymbol{f}_z\boldsymbol{h}_{s'} & -\boldsymbol{f}_z\boldsymbol{h}_{x'} \end{bmatrix} \begin{bmatrix} \boldsymbol{P} \\ \boldsymbol{CP} \end{bmatrix} = \begin{bmatrix} \boldsymbol{g}_s & \boldsymbol{g}_x \\ \boldsymbol{f}_s + \boldsymbol{f}_z\boldsymbol{h}_s & \boldsymbol{f}_x + \boldsymbol{f}_z\boldsymbol{h}_x \end{bmatrix} \begin{bmatrix} \boldsymbol{I} \\ \boldsymbol{C} \end{bmatrix}.$$

The solutions $\boldsymbol{C}$ and $\boldsymbol{P}$ are obtained by the QZ decomposition of the constant matrices and the system can be written as

$$\boldsymbol{Q} \begin{bmatrix} \boldsymbol{S}_{11} & \boldsymbol{S}_{12} \\ \boldsymbol{0} & \boldsymbol{S}_{22} \end{bmatrix} \begin{bmatrix} \boldsymbol{Z}_{11}^H & \boldsymbol{Z}_{21}^H \\ \boldsymbol{Z}_{12}^H & \boldsymbol{Z}_{22}^H \end{bmatrix} \begin{bmatrix} \boldsymbol{P} \\ \boldsymbol{CP} \end{bmatrix} = \boldsymbol{Q} \begin{bmatrix} \boldsymbol{T}_{11} & \boldsymbol{T}_{12} \\ \boldsymbol{0} & \boldsymbol{T}_{22} \end{bmatrix} \begin{bmatrix} \boldsymbol{Z}_{11}^H & \boldsymbol{Z}_{21}^H \\ \boldsymbol{Z}_{12}^H & \boldsymbol{Z}_{22}^H \end{bmatrix} \begin{bmatrix} \boldsymbol{I} \\ \boldsymbol{C} \end{bmatrix}$$

where $\boldsymbol{Q}$ and $\boldsymbol{Z}$ are unitary, $\boldsymbol{S}$ and $\boldsymbol{T}$ are upper triangular matrices and H denotes the conjugate transpose. To ensure stability for the equilibrium process the decomposition the eigenvalues smaller than 1 in absolute value are placed in the left upper corner, see for example Blanchard and Kahn (1980) or Klein (2000). The eigenvalues are obtained by element wise ratios of the diagonal elements of $\boldsymbol{T}$ and $\boldsymbol{S}$. Using the properties $\boldsymbol{Z}_{11}^H\boldsymbol{Z}_{11} + \boldsymbol{Z}_{21}^H\boldsymbol{Z}_{21} = \boldsymbol{I}$ and $\boldsymbol{Z}_{12}^H\boldsymbol{Z}_{11} + \boldsymbol{Z}_{22}^H\boldsymbol{Z}_{21} = \boldsymbol{0}$ of unitary matrices the policy function is given by

$$\boldsymbol{C} = \boldsymbol{Z}_{21}\boldsymbol{Z}_{11}^{-1}$$

and the implied state transition by

$$\boldsymbol{P} = \boldsymbol{Z}_{11}\boldsymbol{S}_{11}^{-1}\boldsymbol{T}_{11}\boldsymbol{Z}_{11}^{-1}. \tag{12}$$

These solutions can be used to either simulate the system or calculate the likelihood of the model. The matrices $\boldsymbol{C}$ and $\boldsymbol{P}$ are only the slope coefficients of the policy function and equilibrium dynamics. The constants are given by the steady state values and the complete functions are

$$\begin{aligned} \boldsymbol{x}_{t+1}^* &= \bar{\boldsymbol{x}} + \boldsymbol{C}(\boldsymbol{s}_t - \bar{\boldsymbol{s}}) \\ \boldsymbol{s}_{t+1}^* &= \bar{\boldsymbol{s}} + \boldsymbol{P}(\boldsymbol{s}_t - \bar{\boldsymbol{s}}). \end{aligned}$$

The linearized measurement equation of the state space form is

$$\begin{aligned} \boldsymbol{m}(\boldsymbol{s}, \boldsymbol{x}) &\approx \bar{\boldsymbol{m}} + \bar{\boldsymbol{m}}_s(\boldsymbol{s} - \bar{\boldsymbol{s}}) + \bar{\boldsymbol{m}}_x(\boldsymbol{x} - \bar{\boldsymbol{x}}) \\ &\equiv \bar{\boldsymbol{m}} + \bar{\boldsymbol{m}}_s(\boldsymbol{s} - \bar{\boldsymbol{s}}) + \bar{\boldsymbol{m}}_x(\bar{\boldsymbol{x}} + C(\boldsymbol{s} - \bar{\boldsymbol{s}}) - \bar{\boldsymbol{x}}) \\ &= \bar{\boldsymbol{m}} + (\bar{\boldsymbol{m}}_s + \bar{\boldsymbol{m}}_x C)(\boldsymbol{s} - \bar{\boldsymbol{s}}). \end{aligned}$$

where $\bar{\boldsymbol{m}}_{..} \equiv \boldsymbol{m}_{..}(\bar{\boldsymbol{s}}, \bar{\boldsymbol{x}})$ denotes the measurement function and its Jacobians evaluated at the steady state. The slope of the linearized measurement equation is therefore

$$\boldsymbol{M} = \bar{\boldsymbol{m}}_s + \bar{\boldsymbol{m}}_x\boldsymbol{C} \tag{13}$$

and the constant term

$$\bar{\boldsymbol{M}} = \bar{\boldsymbol{m}} - \boldsymbol{M}\bar{\boldsymbol{s}} \tag{14}$$

where $\bar{\boldsymbol{m}} = \bar{\boldsymbol{y}}$ is the steady state of the observables.

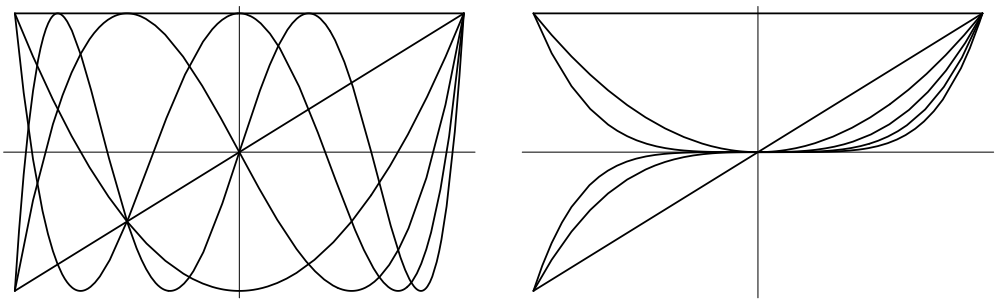
3.3.2 Spectral

Spectral methods use basis functions which are nonzero almost everywhere in the approximation hyper cube $S \subset \mathbb{R}^{d_s}$. Basis functions are orthogonal polynomials like Chebyshev polynomials defined by

$$T_0(s) = 1 \qquad T_1(s) = s \qquad T_{i+1}(s) = 2s\,T_i(s) - T_{i-1}(s) \qquad \text{for } i = 1, ...$$

The problem of monomials $\psi_i(s) = s^i$ compared to orthogonal polynomials is similar to multicollinearity in regression analysis: information is poorly used if regressors are correlated. Monomials are highly correlated and orthogonal polynomials by definition are not. Figure 1 shows polynomials of rising degrees. The orthogonal Chebyshev polynomials in the left figure fill the space more uniformly than monomials and result in higher approximation quality. An approximating

Figure 1: Chebyshev Polynomials and Monomials



of a function by the collocation method requires the approximation to be exact at some chosen points. It is compared to other methods in section 3.6.
The approximation of function $f(s)$ is given by $\hat{f}(s) = \sum_{i=0}^{n} c_i T_i(s)$. To identify the $n+1$ parameters $\boldsymbol{c} = [c_0\ c_1\ c_2]'$ we have to evaluate the function at $n+1$ points. The $n+1$ collocation conditions exactly determine the solution vector $\boldsymbol{c}$

$$\boldsymbol{T}(\boldsymbol{s})\,\boldsymbol{c} = f(\boldsymbol{s}) \tag{15}$$

$$\begin{bmatrix} T_0(s_0) & T_1(s_0) & T_2(s_0) \\ T_0(s_1) & T_1(s_1) & T_2(s_1) \\ T_0(s_2) & T_1(s_2) & T_2(s_2) \end{bmatrix} \begin{bmatrix} c_0 \\ c_1 \\ c_2 \end{bmatrix} = \begin{bmatrix} f(s_0) \\ f(s_1) \\ f(s_2) \end{bmatrix}$$

The basis matrix $\boldsymbol{T}(\boldsymbol{s})$ in equation (15) is a constant for given $\boldsymbol{s}$. The orthogonality property of Chebyshev polynomials ensures that this matrix has a low condition number. It can therefore be inverted without numerical problems in a finite precision environment as opposed to a basis matrix of monomials. Left multiplication of the function values with the inverse basis matrix determines the solution parameters $\boldsymbol{c} = \boldsymbol{T}^{-1}(\boldsymbol{s})\,f(\boldsymbol{s})$.

The collocation approach is not interesting for econometrics since $n+1$ data points $f(\boldsymbol{s})$ are translated into $n+1$ parameters $\boldsymbol{c}$ and the statistical goal of dimension reduction is not achieved. An overdetermined system with more points than parameters can be solved for the parameters by least squares.
How should the points $s_0, ..., s_n$ be chosen? The numerical theory says that the optimal points or nodes are the roots of a Chebyshev polynomial. Fortunately there is an explicit solution for Chebyshev polynomial roots given by

$$s_k^T = \cos\left(\frac{\pi(k+0.5)}{n+1}\right) \qquad \text{for } k = 0, ..., n.$$

In the three point example the solutions for $T_3(x) = 4x^3 - 3x = 0$ are $(0, \pm\sqrt{3}/2)$. The solutions are always within the interval $(-1, 1)$ and have to be transformed linearly if another approximation interval is needed. If the approximation interval is $S = [a, b]$ the appropriate nodes are $s_k = a + (b-a)(s_k^T + 1)/2$ for $k = 0, ..., n$. The inverse transformation has to be applied if interpolated function values at other points than s_k are needed. The approximation coefficients are determined with the basis matrix evaluated at the Chebyshev roots $\boldsymbol{s}^T = [s_0^T, ..., s_n^T]$ whereas the function is evaluated at the transformed points $\boldsymbol{s} = [s_0, ..., s_n]$

$$\boldsymbol{c} = \boldsymbol{T}^{-1}(\boldsymbol{s}^T) f(\boldsymbol{s}).$$

For an interpolation aside the nodes at x the inverse linear transformation $x^T = 2(x-a)/(b-a) - 1$ has to be applied where the basis matrix is evaluated at

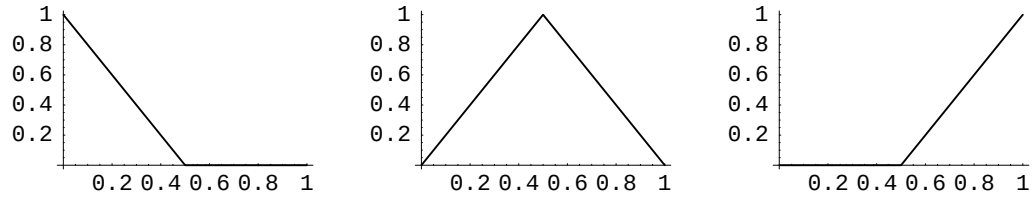
$$\hat{f}(x) = \boldsymbol{T}(x^T)\,\boldsymbol{c}.$$

The complication in economic models is that the policy functions we want to approximate are unknown and we cannot obtain function values at the nodes $f(\boldsymbol{s})$ by simple evaluations. Instead we have to guess start values and improve them according to an accuracy measure. The accuracy is measured by the residuals of the first order conditions when states and policies are plugged in. The residuals should be small since for the exact policy function they are zero for all possible states. The start policy values will usually not provide sufficiently small residuals and some methods are needed to change them in the direction where the residuals approach zero. This problem will be discussed in section 3.7. The second problem is that rational expectation models involve an expectation operator in the functional equations and therefore some integrals have to be evaluated before the residual is obtained. This will be discussed in section 3.4.

3.3.3 Finite Elements

The principle of finite elements is to divide the approximation space into many subspaces where low degree basis functions are used for approximation. Figure

Figure 2: Linear Finite Elements



2 shows three linear local basis functions known as hat functions in a one-dimensional approximation problem. The approximation space is divided into two intervals $[0, .5]$ and $[.5, 1]$ by three nodes $\{0, .5, 1\}$. There are always two basis functions which are not zero in each interval. In the first interval the right wing of the hat function with peak at 0 and the left wing of the hat function with peak at 0.5 are not zero. If we want the function to be exactly approximated at the nodes (collocation approach, see section 3.6) we need two parameters per interval. These parameters determine the linear combination of the basis functions. They are equal to the function values at the nodes where one basis functions is 1 and the other 0. Therefore the collocation approach with linear splines amounts to solve the nonlinear functional at some nodes. The evaluation of the approximation at other points than the nodes is then a simple linear interpolation in the appropriate interval. As opposed to the spectral approximation the nodes can be chosen freely.
In integral equations linear finite elements can be used to approximate the integrand. Once this is done the integral can be easily approximated as the area under the linear functions. Higher order local polynomials should be used if first derivatives have to be smooth since with linear finite elements they have kinks at the nodes.

3.4 Integration

Rational expectations in economic models are represented by integration operators. Closed form integrals for nonlinear models are usually unknown and numerical integration is based on deterministic or by random methods.
Monte Carlo integration chooses random points in the interval of integration evaluates the integrand. The average is an estimate of the expected value of the function. Since the integral represents the area under the integrand, the average multiplied by the size of the interval is an estimate of the integral. The usual argument for a Monte Carlo approach is that it is independent of the problem dimension. This is at best misleading since it is the convergence rate which is independent and moreover rather low. The problem with the Monte Carlo approach is that it is often much too general in the sense that smoothness of the

integrand is not exploited whereas most economic models are described in terms of smooth functions. Smoothness can be described by bounded mixed derivatives up to order r.
In order to achieve an accuracy ϵ, the convergence for Monte Carlo methods is described by

$$\epsilon(n) = \mathcal{O}(n^{1/2}).$$

It is independent of dimension d and smoothness r, see for example Gerstner and Griebel (2003). Random numbers in binary computers are not really random but generated by deterministic sequences which mimic some important random number properties. For the purpose of integration better sequences can be generated which fill the space more uniformly like Niederreiter sequences. They reduce complexity to

$$\epsilon(n) = \mathcal{O}(n^{-1}(\log n)^d).$$

An integration technique which exploits smoothness of the integrand is Gaussian quadrature. Its complexity is

$$\epsilon(n) = \mathcal{O}(n^{-r/d})$$

and the amount of work to achieve a prescribed accuracy grows exponentially with the dimension d. This exponential dependency is known as the curse of dimensionality. The rule of thumb for Kronecker Gaussian quadrature is that it is not feasible for integrals with a higher dimension than four or five.
The complexity of Smolyak Gaussian quadrature is of order

$$\epsilon(n) = \mathcal{O}(n^{-1}(\log n)^{(d-1)(r+1)})$$

and can be expected to outperform Quasi-Monte Carlo methods for smooth functions of order $r > 1$. For very smooth integrands with $r \to \infty$ convergence is even exponential. This method does not suffer from the exponential curse of dimensionality.
Gaussian quadrature was applied in economics for example in Tauchen and Hussey (1991). Univariate quadrature chooses some deterministic points x_i and weights w_i and takes a weighted mean of the integrand as the integral approximation

$$\int g(x)\,dx \approx \sum_{i=0}^{n-1} w_i g(x_i).$$

The evaluation of the integrand is the most costly part of numerical integration and the amount of work can be measured by the number of nodes n.

A useful starting point to understand how these nodes and weights are derived is to approximate the integrand g by a polynomial

$$g(x) \approx \sum_{i=0}^{n-1} c_i x^i.$$

Quadrature works well for smooth integrands where a small number of nodes and therefore integrand evaluations is sufficient for an accurate approximation. Once the approximating polynomial is obtained, the integral value can be given in closed form

$$\int_a^b g(x)\,dx \approx \int_a^b \sum_{i=0}^{n-1} c_i x^i dx = \sum_{i=0}^{n-1} \int_a^b c_i x^i dx = \sum_{i=0}^{n-1} \left[c_i \frac{x^{i+1}}{i+1} \right]_a^b = \sum_{i=0}^{n-1} c_i \frac{b^{i+1} - a^{i+1}}{i+1}.$$

It is a simple function of the coefficients c_i, the polynomial degree n and the integration bounds a, b. There is no fundamental difference between integration and approximation operators and the Smolyak operator works equally for both. Both operators use function evaluations at some nodes x_i. Whereas function approximation delivers the polynomial coefficients c_i, quadrature derives the weights w_i. The weights are in fact a function of the coefficients of the polynomial which approximates the integrand.
If the function g can be separated into $g(x) = f(x)\,w(x)$ further efficiency can be gained. Gaussian quadrature is the general approach and specialized formulas derive nodes and weights for a given weight function w

$$\int g(x)\,dx = \int f(x)\,w(x)\,dx \approx \sum_{i=1}^{n} w_i\,f(x_i).$$

In general an appropriate family of orthogonal polynomials can be derived for many weight functions which can then be used to approximate the integral, see Press, Flannery, Teukolsky, and Vetterling (1988). For standard weight functions the orthogonal polynomials are known. For example Legendre polynomials, associated with $w(x) = 1$, $\forall x$, are used for an unweighted integration with $g = f$. The estimation of structural parameters as well as rational expectations involve the weighted form of integration where the weight function is a density. In usual economic models Hermite polynomials can be used for the kernel $w(x) = e^{-x^2}$ of a standard normal distribution.
The integral of a density weighted function is the expected value of the transformed random variable

$$\mathrm{E}(f(x)) = \int f(x)\,p\,(x)\,dx.$$

In principle one can integrate the combined function $g(x) = f(x)\,p\,(x)$ by Gauss-Legendre quadrature. Instead of approximating a density function p by a polynomial it is more efficient to represented it by a collection of nodes and associated

weights. In the limit of an infinite number of nodes, the weighted sum of integrand values at the nodes is the expected value of interest. Gaussian quadrature of a density weighted function is therefore nothing else than a clever way to discretize the continuous random variable x. The integral is approximated by a weighted sum of integrand evaluations at the nodes. The discrete density approximation is a histogram with n nodes x_i and weights w_i

$$p\,(x) \approx \sum_{i=1}^{n} w_i \delta(x_i - x).$$

The Dirac delta function δ is defined by $\int f(x)\,\delta(x-a)\,dx = f(a)$ and allows to combine the infinitesimal integration operator and its discrete counterpart. The discrete approximation of a density, substituted in the integral functional gives

$$\begin{aligned} \mathrm{E}(f(x)) &\approx \int f(x) \sum_{i=1}^{n} w_i \delta(x_i - x)\,dx \\ &= \sum_{i=1}^{n} w_i f(x_i). \end{aligned}$$

Higher moments of a transformed random variable are expected values of the function raised to higher powers and the same procedure applies to obtain them. This will be needed in the estimation part in section 4.
How to choose the nodes and weights for a given density p? As for Chebyshev function approximation the nodes are given by the roots of the involved polynomial. For a normal density the associated Hermite polynomials are given by the recursion

$$H_0(x) = 1 \qquad H_1(x) = 2x \qquad H_{n+1}(x) = 2xH_n(x) - 2nH_{n-1}(x).$$

Unfortunately a closed form solution for larger n does not exist and a nonlinear root finder is needed. Press, Flannery, Teukolsky, and Vetterling (1988) give good start values for different densities such that a simple Newton iteration finds the roots. For a three point approximation of a standard normal density the nodes are $\{-\sqrt{3/2}, 0, \sqrt{3/2}\}$.
The criterium to derive the weights is rather obvious: the moments of the discrete variable have to be equal to the moments of the original continuous variable. The number of matched moments depends on the number of nodes and the moment matching condition is given by

$$\int_a^b H_k(x)\,p\,(x)\,dx = \sum_{i=1}^{n} \omega_i\, H_k(x_i) \qquad \text{for } k = 0, \ldots, n-1. \tag{16}$$

For $k = 0$ the Hermite polynomial is 1 and the first condition requires the weights to sum to 1 as an obvious need for a univariate density approximation. For $k = 1$

the Hermite polynomial is $2\,x$ and the condition matches the first raw moment. For $k = 2$ the second is matched and so on. Another interesting interpretation of the moment matching condition is related to the polynomial approximation viewpoint of integration. In equation (16) we can see that the expected value of a polynomial transformation will be exact. The number of nodes determines for which maximal degree of polynomial transformation the approximation of the expected value will be exact. Or put it differently, if a high degree polynomial is needed for an approximation of the integrand then its integration will need many nodes. The moment matching condition can be further simplified to

$$\begin{aligned} \int_a^b p\,(x)\,dx = 1 &= \omega_1 + \cdots + \omega_n && \text{for } k = 0 \text{ by } H_0(x) = 1 \\ 0 &= \omega_1 H_k(x_1) + \cdots + \omega_n H_k(x_n) && \text{for } k > 0 \text{ by } \langle H_k, H_0 \rangle = 0 \end{aligned}$$

and summarized in a linear system for the weights $\boldsymbol{\omega}$

$$\begin{array}{ccc} \boldsymbol{\Psi} & = & \boldsymbol{\Phi}\,\boldsymbol{\omega} \\ \begin{pmatrix} 1 \\ 0 \\ \vdots \end{pmatrix} & = & \begin{pmatrix} H_0(x_1) & \cdots & H_0(x_n) \\ \vdots & \ddots & \vdots \\ H_{n-1}(x_1) & \cdots & H_{n-1}(x_n) \end{pmatrix} \begin{pmatrix} \omega_1 \\ \vdots \\ \omega_n \end{pmatrix}. \end{array}$$

The problem with Gaussian quadrature is that it is defined for univariate integrals and it is silent about the extension of the operator to the multivariate case. The obvious extension is to match the moments of the multivariate random variable and the question is how to combine nodes in one dimension to obtain nodes in many dimensions. The simplest approach is to discretize each of the involved random variables separately and combine the univariate nodes and weights by a Kronecker product rule. This and the Smolyak extension are described in section 3.5.

The rational expectations in the general model are formed over the integrand $\boldsymbol{h}$. $p\,(x)$ is then a multivariate density of a vector of random shocks $\boldsymbol{e}'$ and the integral is

$$\begin{aligned} \boldsymbol{z} &= \int \boldsymbol{h}(\boldsymbol{s}, \boldsymbol{x}, \quad \boldsymbol{e}', \boldsymbol{s}', \qquad\qquad \boldsymbol{x}' \qquad\qquad))\, p\,(\boldsymbol{e}')\, d\boldsymbol{e}' \\ &= \int \boldsymbol{h}(\boldsymbol{s}, \boldsymbol{x}^*(\boldsymbol{s}), \boldsymbol{e}', \boldsymbol{g}(\boldsymbol{s}, \boldsymbol{x}^*(\boldsymbol{s}), \boldsymbol{e}'), \boldsymbol{x}^*(\boldsymbol{g}(\boldsymbol{s}, \boldsymbol{x}^*(\boldsymbol{s}), \boldsymbol{e}'))\, p\,(\boldsymbol{e}')\, d\boldsymbol{e}'. \end{aligned}$$

3.5 Operators

Usually the functions we are approximating or integrating depend on several variables. Therefore we need a rule to extend univariate operators to many dimensions. The usual and most simple approach is to do it by the Kronecker operator. It combines each of the univariate nodes and basis functions with all the others.

An intuition about how to deal with that exponentially growing computational burden more carefully is to take a look at a Taylor series expansion of a function. A second order expansion of a bivariate function around (x^*, y^*) is given by

$$\begin{aligned} F(x,y) &\approx F(x^*,y^*) \\ &+ F_x(x^*,y^*)(x-x^*) + F_y(x^*,y^*)(y-y^*) \\ &+ \frac{1}{2}[F_{xx}(x^*,y^*)(x-x^*)^2 + 2F_{xy}(x^*,y^*)(x-x^*)(y-y^*) \\ &+ F_{yy}(x^*,y^*)(y-y^*)^2] \end{aligned}$$

It can be further simplified by combining the constant terms x^*, y^* and $F_{..}(x^*, y^*)$

$$F(x,y) \approx \alpha_0 + \alpha_1 x + \alpha_2 y + \alpha_3 xy + \alpha_4 x^2 + \alpha_5 y^2.$$

This second order bivariate Taylor expansion is not derived by the Kronecker product of second order univariate polynomials. This would be a linear combination of the polynomials

$$\{1, x, y, xy, x^2, y^2, x^2y, xy^2, x^2y^2\} = \{1, x, x^2\} \otimes \{1, y, y^2\}.$$

The important message is that we can drop the elements x^2y, xy^2, x^2y^2 without loosing asymptotic accuracy. The resulting polynomials are called complete polynomials and are characterized by the property that terms only up to a certain sum of the powers of the involved univariate polynomials are used. In the example we need only basis functions of the Kronecker product $x^a y^b$ where $0 \leq a + b \leq 2$.
An important questions for approximation and integration is how to combine nodes in one dimensions to form nodes in many dimensions. For Kronecker operations on functions the associated operation on the one dimensional nodes is the simple Cartesian product. The optimal multidimensional nodes for complete polynomials are not obvious without the Smolyak operator. Since the operator gives complete polynomials as a special case it can also be used to derive the associated optimal grid. The principle of using only some combinations of Kronecker products was formulated as early as Smolyak (1963). It was recently used by Krüger and Kübler (2004) to solve an OLG model.
The operator also extends finite element approximation of any degree to many dimensions. Since function approximation and integration are fundamentally the same operators, the Smolyak operator can be applied to both operators without being subject to the curse of dimensionality.
Another very useful property of the Smolyak algorithm is that it constructs the approximation hierarchically. This feature gives an approximation accuracy estimate as a by-product and will be discussed in section 3.5.3.

3.5.1 Kronecker

The univariate ($d = 1$) approximation operator for a function $f : [0, 1] \to \mathbb{R}$ is

$$U^i(f) = \sum_{j=1}^{m_i} a_j^i \, f(x_j^i)$$

where $i \in \mathbb{N}$, $a_j^i \in \mathbb{C}([-1, 1])$ and $x_j^i \in [-1, 1]$ for a normed approximation. In case of approximating an integration operator, a_j^i are the weights and therefore real numbers and in case of function approximation they are functions implicitly containing the polynomial coefficients. The multidimensional ($d > 1$) tensor or Kronecker product operator $\otimes$ is given by

$$(U^{i_1} \otimes \cdots \otimes U^{i_d})(f) = \sum_{j_1=1}^{m_{i_1}} \cdots \sum_{j_d=1}^{m_{i_d}} (a_{j_1}^{i_1} \otimes \cdots \otimes a_{j_d}^{i_d}) f(x_{j_1}^{i_1}, ..., x_{j_d}^{i_d}).$$

In this algorithm $m_{i_1} \cdots m_{i_d}$ function evaluations are needed to construct the approximation or integration. This establishes the curse of dimensionality of the Kronecker operator.

3.5.2 Smolyak

The Smolyak operator is an active topic in numerics and especially complexity research, see for example Gerstner and Griebel (2003), Novak and Ritter (1999) or Wasilkowski and Woźniakowski (1999). It is a recursive construction given by

$$A_{q,d}(f) = \sum_{|\,\mathbf{i}\,| \le q} (\Delta^{i_1} \otimes \cdots \otimes \Delta^{i_d})(f) = A_{q-1,d}(f) + \underbrace{\sum_{|\,\mathbf{i}\,| = q} (\Delta^{i_1} \otimes \cdots \otimes \Delta^{i_d})(f)}_{\Delta A_{q,d}(f)} \quad (17)$$

where $A_{d-1,d} = 0$, $U^0 = 0$, $\Delta^i = U^i - U^{i-1}$. $\mathbf{i}$ is a d-dimensional vector with norm $|\,\mathbf{i}\,| = i_1 + \cdots + i_d$. $q \ge d$ drives the approximation accuracy. This formulation does not repeat polynomial terms in the various U^i and is more suited for programming purposes. The following formulation reveals the underlying structure more clearly

$$A_{q,d}(f) = \sum_{q-d+1 \le |\,\mathbf{i}\,| \le q} (-1)^{q-|\,\mathbf{i}\,|} \binom{d-1}{q-|\,\mathbf{i}\,|} (U^{i_1} \otimes \cdots \otimes U^{i_d}). \quad (18)$$

The Smolyak operator is a linear combination of certain low level Kronecker product operators. The recursive structure in equation (17) suggests that the approximation can be refined by increasing q. At each level the approximation gain can be calculated. In case of a function approximation the gain is given by the difference between the interpolation and the true function value at the finer grid. In case of an integration the accuracy gain is simply the change of

the integral value. Both values, if small enough, can be used to stop at a certain accuracy before a maximal approximation degree is reached.
Another rule to reduce the curse of dimensionality is the Gaussian quadrature algorithm of Genz and Keister (1996). It is an application of the fully symmetric method of Genz (1986) to Patterson (1968) extensions of Gaussian quadrature for normally distributed shocks. Novak and Ritter (1999) show that the fully symmetric algorithm is a special case of the Smolyak operator. I use it for the rational expectation and filtering integrals since Patterson (1968) extensions use nested univariate nodes and thus allow for adaptive integration. Any other nonnested quadrature scheme can be used for nonadaptive integration. Heiss and Winschel (2005) demonstrated the operator's potential in microeconometrics where Smolyak Gaussian quadrature of nonlinear likelihood integrals in a mixed logit model dramatically outperformed Monte Carlo methods up to twenty dimensions. The operator establishes Gaussian quadrature as a competitor to simulation methods for high dimensions and smooth functions. Common wisdom was that integrals beyond five dimensions can be solved only by Monte Carlo methods. The basic operator is competitive at least up to 30 dimensions. Further refinement is possible and integrals with a special structure and several hundred dimensions were solved in Gerstner and Griebel (2003).
The Smolyak rule operates on a set of one dimensional nodes analogously to operations on functions in order to construct the multidimensional nodes. These nodes are the points where f is evaluated in order to construct $A_{q,d}(f)$. The one dimensional nodes $X^{i_1},..,X^{i_d}$, $X^i = \{x_1^i,...,x_{m_i}^i\}$, used by U^i, are not combined by the Cartesian product $\times$. Instead again only some certain combinations are used

$$H_{q,d} = \bigcup_{q-d+1 \leq |\mathbf{i}| \leq q} \left(X^{i_1} \times \cdots \times X^{i_d}\right). \tag{19}$$

For a recursive approximation refinement, the function evaluations in one level should be reused in the next level. The one dimensional nodes have then to be selected in a nested way so that $X^i \subset X^{i+1}$. This implies $(X^i \times X^j) \subset (X^i \times X^{j+1})$ and therefore

$$H_{q,d} \subset H_{q+1,d}. \tag{20}$$

Equation (19) can be rewritten recursively

$$H_{q,d} = \bigcup_{|\mathbf{i}| \leq q} \left(X_\Delta^{i_1} \times \cdots \times X_\Delta^{i_d}\right) \tag{21}$$

$$\Delta H_{q,d} = \bigcup_{|\mathbf{i}| = q} \left(X_\Delta^{i_1} \times \cdots \times X_\Delta^{i_d}\right) \tag{22}$$

$$H_{q,d} = H_{q-1,d} \cup \Delta H_{q,d}. \tag{23}$$

with $X^0 = \emptyset$, $X_\Delta^i = X^i \setminus X^{i-1}$ and $H_{d-1,d} = \emptyset$.

The grid used in this work is the Gauss-Lobatto grid defined by

$$m_i = \begin{cases} 1 & \text{for } i = 1 \\ 2^{i-1} + 1 & \text{for } i > 1 \end{cases}$$

$$x_j^i = \begin{cases} 0 & \text{for } i = 1 \\ -\cos\left(\frac{\pi(j-1)}{m_i - 1}\right) & \text{for } j = 1, ..., m_i. \end{cases}$$

These nodes are different than the roots of Chebyshev polynomials. They deliver similar optimal results but have the advantage to be nested.
The following examples are meant to clarify the unusual notation and therefore apparently complicated structure of the Smolyak formulas. The key is the summation set of the index vectors $\mathbf{i}$ in formula (18). In case of a two dimensional ($d = 2$) operator these are the following sets of vectors

$$\begin{aligned} q = 2 : \{\mathbf{i} : 2 - 2 + 1 = 1 \leq |\,\mathbf{i}\,| \leq 2\} &= \{[1\ 1]\} \\ q = 3 : \{\mathbf{i} : 3 - 2 + 1 = 2 \leq |\,\mathbf{i}\,| \leq 3\} &= \{[1\ 1], [1\ 2], [2\ 1]\} \\ q = 4 : \{\mathbf{i} : 4 - 2 + 1 = 3 \leq |\,\mathbf{i}\,| \leq 4\} &= \{[1\ 2], [2\ 1], [1\ 3], [3\ 1], [2\ 2]\} \end{aligned}$$

This gives the following two dimensional Chebyshev polynomials in x and y for the levels $q = 2$ and $q = 3$

$$\begin{aligned} q = 2 \quad : \quad & \{(c_1^1) \times (c_1^2)\} \\ q = 3 \quad : \quad & \{(c_1^1) \times (c_1^2)\} \\ & \{(c_1^1) \times (c_1^2 + c_2^2 y + c_3^2 (2y^2 - 1))\} \\ & \{(c_1^1 + c_2^1 x + c_3^1 (2x^2 - 1)) \times (c_1^2)\}. \end{aligned}$$

The recurring polynomial terms are left out at subsequent levels in the representation in equation (17).
The Smolyak operator on the one dimensional nodes

$$\begin{aligned} &i = 1,\ m_i = 1,\ X^i = \{0\} \\ &i = 2,\ m_i = 3,\ X^i = \{-1, 0, 1\} \\ &i = 3,\ m_i = 5,\ X^i = \{-1, -2^{-0.5}, 0, 2^{-0.5}, 1\}. \end{aligned}$$

constructs the first level of a two dimensional grid as

$$\begin{aligned} H_{2,2} &= (X^1 \times X^1) \\ &= [0\ 0]. \end{aligned}$$

Figure 3: Gauss-Lobatto grid at Levels k

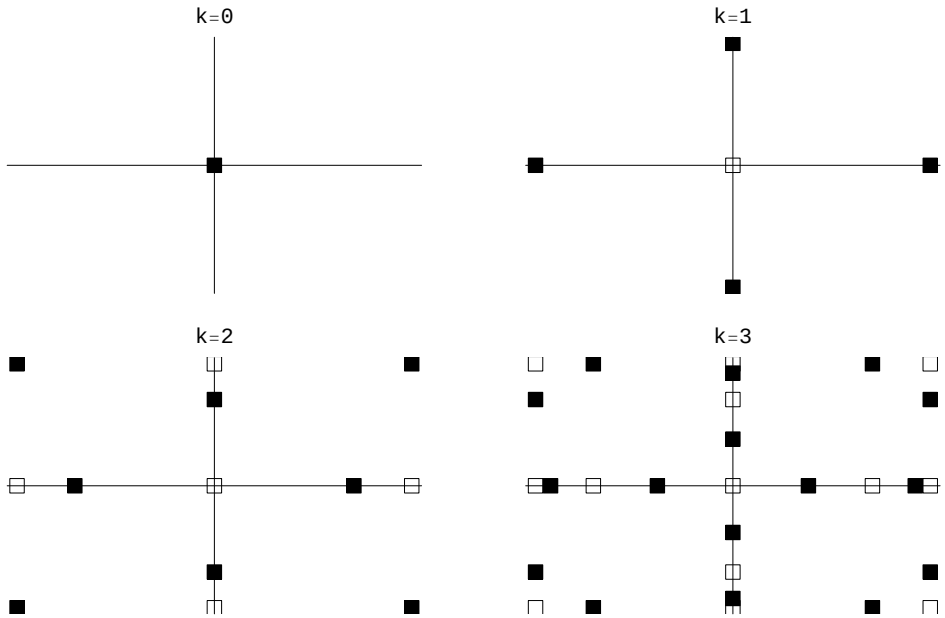
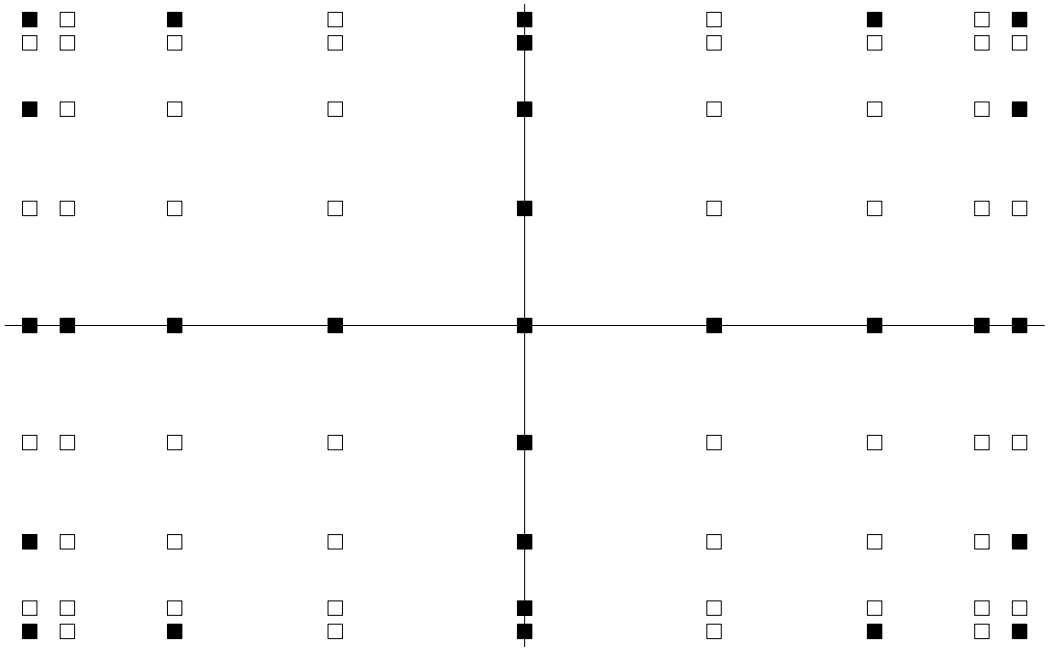


Figure 4: Gauss-Lobatto versus Kronecker Product Grid



The grid for the next level is

$$\begin{aligned}
H_{3,2} &\overset{(19)}{=} (X^1 \times X^1) \cup (X^1 \times X^2) \cup (X^2 \times X^1) \\
&\overset{(20)}{=} (X^1 \times X^2) \cup (X^2 \times X^1) \\
&= \{[0\ -1], [0\ 0], [0\ 1]\} \cup \{[-1\ 0], [0\ 0], [1\ 0]\} \\
&\overset{(21)}{=} (X^1_\Delta \times X^1_\Delta) \cup (X^1_\Delta \times X^2_\Delta) \cup (X^2_\Delta \times X^1_\Delta) \\
&\overset{(23)}{=} (X^1 \times X^1) \cup (X^1_\Delta \times X^2_\Delta) \cup (X^2_\Delta \times X^1_\Delta) \\
&= \{[0\ 0]\} \cup \{[0\ -1], [0\ 1]\} \cup \{[-1\ 0], [1\ 0]\}
\end{aligned}$$

where the numbers over the equality signs refer to the applied equation. The node [0 0] is contained in the first level grid and appears for nested univariate nodes also in the second level grid. The difference form in equation (21) and (23) avoids this recurrence and constructs only additional nodes for each level. The grid for the next level is given by

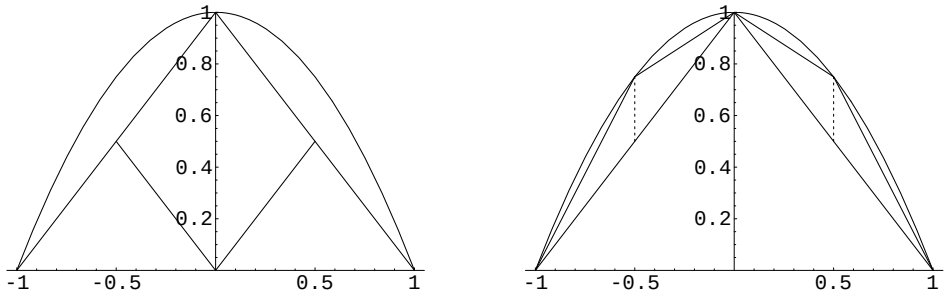
$$H_{4,2} = (X^1 \times X^3) \cup (X^2 \times X^2) \cup (X^3 \times X^1).$$

Figure 3 shows the grid for level $k \equiv q - d = 0, 1, 2, 3$ where the filled squares are the new points of the level. Figure 4 contrasts the Gauss-Lobatto grid to the tensor product grid.

3.5.3 Adaptivity

An interesting feature of the recursive or hierarchical nature of the Smolyak operator is that a convergence criterion is available almost for free. The intuition can be seen most easily in Archimedes' approximation strategy for $f(x) = 1 - x^2$ in figure 5. This strategy we would call today hierarchical linear splines.
In the left picture the first and the second level basis functions are plotted together with the function to be approximated. The right hand side shows the linear combination of both levels' basis functions. The two dotted vertical lines are the ap-

Figure 5: Hierarchical Surpluses



proximation gains called hierarchical surpluses. They are constructed by approximating the function on the coarse grid $H_{1,1} = \{[-1], [0], [1]\}$ and interpolating the function values at the additional points of the finer grid $\Delta H_{2,1} = \{[-0.5], [0.5]\}$. The difference between these interpolations and the true function values at $\Delta H_{2,1}$ is the surplus. The convergence of the surpluses to zero for ever finer grids makes them a natural convergence criterion. Therefore it is possible to construct algorithms with a stopping criterion once a specified accuracy level is reached. Some structural parameter vectors may imply more or less linear policy functions and an à priori specified level of approximation is inefficient.
This kind of adaptivity approximates up to a certain accuracy for all dimensions. It is already an integral part of the Smolyak operator due to its hierarchical structure. I have implemented it for the integration operator. It is rather simple for nested univariate nodes since the Smolyak algorithm adds in each level some new nodes where the integrand has to be evaluated. The evaluations at lower level nodes can be reused and have only to be reweighted.
There is another more complicated adaptive scheme discussed by Gerstner and Griebel (2003) which approximates with different accuracy in each dimension. They approximate a 360 dimensional integral in two minutes on a 400 MHz Pentium machine. One of their examples is from economics where a collateral mortgage obligation is priced by a present value integral. The next period realization is discounted only once whereas later realization are discounted more often and contribute less to the present value. It is therefore economical to approximate the next period realization more accurate than the realization in the more distant future. This dimension adaptive approximation scheme is not implemented in the current code.

3.6 Weighted Residuals

According to McGrattan (1998) there are three broad approaches to identify the parameter $\boldsymbol{c}$ of the approximated policy function. Common to all of them is that the integral of weighted residuals over the state has to be zero. The methods are characterized by different weight functions.
The residual in the economic application is the value of the first order condition with the approximated policy functions substituted for the policy variable. For the general model the residual is given by

$$\begin{aligned} \boldsymbol{r}^*(\boldsymbol{s}, \boldsymbol{x}^*) &= \boldsymbol{f}(\boldsymbol{s}, \boldsymbol{x}^*(\boldsymbol{s}), \sum_j w_j \boldsymbol{h}(\boldsymbol{s}, \boldsymbol{x}^*(\boldsymbol{s}), \boldsymbol{e}'_j, \boldsymbol{s}'_j, \boldsymbol{x}^*(\boldsymbol{s}'_j)) \\ \boldsymbol{s}'_j &= \boldsymbol{g}(\boldsymbol{s}, \boldsymbol{x}^*(\boldsymbol{s}), \boldsymbol{e}'_j). \end{aligned}$$

The policy function is defined by its coefficients $\boldsymbol{x}^*(\boldsymbol{s}) \equiv \boldsymbol{x}^*(\boldsymbol{s}; \boldsymbol{c})$ and the residual can alternatively be written as

$$\boldsymbol{r}(\boldsymbol{s}, \boldsymbol{c}) \equiv \boldsymbol{r}^*(\boldsymbol{s}, \boldsymbol{x}^*).$$

This form is more appropriate in order to highlight the implications of the identifying conditions for policy coefficient vector $\boldsymbol{c} = (c_1, ..., c_n)$. The n conditions require the the n coefficients to be chosen such that the weighted residuals integrate over the states to zero

$$\int_{\boldsymbol{S}} w_i(\boldsymbol{s})\, \boldsymbol{r}(\boldsymbol{s}, \boldsymbol{c})\, d\boldsymbol{s} = 0 \qquad \text{for} \qquad i = 1, ..., n. \tag{24}$$

The Galerkin scheme determines the coefficients so that all available structure in the residuals is used. This requires the residuals to be orthogonal to the basis functions and the weights are therefore the basis functions $w_i(\boldsymbol{s}) = \psi_i(\boldsymbol{s})$ in equation (24).
The least squares method requires to minimize

$$\min_{\boldsymbol{c}} \int_{\boldsymbol{S}} \boldsymbol{r}^2(\boldsymbol{s}, \boldsymbol{c})\, d\boldsymbol{s}$$

and the general form (24) is a first order condition for minimal squared residuals with weights $w_i(\boldsymbol{s}) = \partial\, \boldsymbol{r}(\boldsymbol{s}, \boldsymbol{c}) / \partial\, c_i$. For both methods multidimensional integrals have to be evaluated. The Galerkin method is the optimal choice for finite elements whereas the collocation method is used for spectral approximation.
It specifies the weights as $w_i(\boldsymbol{s}) = \delta(\boldsymbol{s} - \boldsymbol{s}_i)$ where δ is again the Dirac delta function. Substituting the weights in the general form we see that the n identification conditions for $\boldsymbol{c}$ forces the residual to be zero at the nodes

$$\int_{\boldsymbol{S}} \delta(\boldsymbol{s} - \boldsymbol{s}_i)\, \boldsymbol{r}(\boldsymbol{s}, \boldsymbol{c})\, d\boldsymbol{s} = \boldsymbol{r}(\boldsymbol{s}_i, \boldsymbol{c}) = 0 \qquad \text{for} \qquad i = 1, ..., n$$

where $\boldsymbol{s}_i$ is one of n multidimensional nodes.

3.7 Implicit Policy Function

So far the discussion of the approximation methods assumed that the function values at the nodes can be obtained by simply evaluating the function we are approximating. This is not the case in functional equations since the function we want to approximate is the one we are looking for. That means we have to approximate functions we do not know but which have to fulfill the first order conditions. The solution to this problem is an iterative procedure with an initial guess and subsequent refinement until the first order condition at the nodes and their associated policy values is zero. The whole policy function then consists of the exact policy values at the nodes and the interpolated policy elsewhere. Solving for the nonlinear rational expectation equilibrium is therefore a nonlinear root finding procedure with an integral evaluation for the rational expectations along the way to evaluate the residuals. In the Galerkin and least squares procedure the root has to be found in the weighted residuals and in case of the collocation approach in the residuals at the nodes.

Table 2: Implicit Function Iteration

0. choose initial policy values $\boldsymbol{x}^{(0)}$ at grid $\boldsymbol{s}^G$
1. approximate policy function $\boldsymbol{c}^{(k)} = \boldsymbol{\Psi}\left(\boldsymbol{s}^G\right)^{-1}\boldsymbol{x}^{(k)}$
2. calculate rational expectations
   (a) for all discrete shock realizations $j = 0, ..., J$
      i. $\boldsymbol{s'}_j = \boldsymbol{g}(\boldsymbol{s}^G, \boldsymbol{x}^{(k)}, \boldsymbol{e'}_j)$
      ii. $\boldsymbol{x'}_j = \boldsymbol{\Psi}(\boldsymbol{s'}_j)\,\boldsymbol{c}^{(k)}$
   (b) weight future realizations $\boldsymbol{z} = \sum_j w_j \boldsymbol{h}(\boldsymbol{s}^G, \boldsymbol{x}, \boldsymbol{e'}_j, \boldsymbol{s'}_j, \boldsymbol{x'}_j)$
3. solve for expectational policy $\boldsymbol{x}^e$ in $\boldsymbol{f}(\boldsymbol{s}^G, \boldsymbol{x}^e, \boldsymbol{z}) = 0$
4. update policy by
   (a) function iteration $\boldsymbol{x}^{(k+1)} = \alpha\,\boldsymbol{x}^{(k)} + (1-\alpha)\,\boldsymbol{x}^{\boldsymbol{e}}$ or
   (b) root finding $\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} - [\partial\,\boldsymbol{r}(\boldsymbol{s}^G, \boldsymbol{x}^{(k)})/\partial\,\boldsymbol{x}]^{-1}\boldsymbol{r}(\boldsymbol{s}^G, \boldsymbol{x}^{(k)})$
5. $k = k + 1$
6. do 1-6 while $\boldsymbol{r}(\boldsymbol{s}^G, \boldsymbol{x}^{(k)}) \neq 0$

I have implemented the collocation approach with two iterative schemes. The first is a fast Newton type root finding algorithm and the second is a robust function iteration. The root finding scheme is a standard Newton iteration given by

$$\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} - \left(\frac{\partial\,\boldsymbol{r}(\boldsymbol{s}, \boldsymbol{x}^{(k)})}{\partial\,\boldsymbol{x}}\right)^{-1} \boldsymbol{r}(\boldsymbol{s}, \boldsymbol{x}^{(k)})$$

where $\boldsymbol{x}^{(k)}$ are the policy function values at the nodes in iteration k. Since the policy function can be either represented by the policy values $\boldsymbol{x}^{(k)}$ or the implied function coefficients $\boldsymbol{c}^{(k)}$ the function iteration can also be done over the coefficient vector. I iterate over the policy values since some coefficients can be close to zero and the algorithm may become unstable.
The steps for the calculation of the approximation parameters or the policy function at the state grid $\boldsymbol{s}^G$ are summarized in box 2. The first step is to start with some policy values at the state grid in step 0. Step 1 calculates the policy coefficients for the policy at the grid. They are needed for the interpolation in step 2 (a) ii where the next period policy at the state realization has to be cal-

culated. Step 2 evaluates the rational expectations. This is done by calculating for each of the possible discrete shocks $\boldsymbol{e}_j$ the next period state in 2(a)i and the next period policy in 2(a)ii. With next period states and policies for all possible discrete shocks we can evaluate the rational expectations $\mathrm{E}\,\boldsymbol{h}$ by weighting the possible future paths in step 2(b). This gives the expectational variables $\boldsymbol{z}$. In step 3 the expectational policy $\boldsymbol{x}^e$ is derived for a given state grid and rational expectation variable $\boldsymbol{z}$ by an inner root finding algorithm. For some models, like the example model, $\boldsymbol{f}(\boldsymbol{s}^G, \boldsymbol{x}^e, \boldsymbol{z}) = 0$ can be solved explicitly for $\boldsymbol{x}^e$ and the inner root finding is not necessary. The optimal policy is found if the expectational policy is the same $\boldsymbol{x}^e = \boldsymbol{x}^{(k)}$ as the one used to generate the expectational variables $\boldsymbol{z}(\boldsymbol{x}^{(k)})$. This is another way to say that the residual for this policy has to be zero $\boldsymbol{f}(\boldsymbol{s}, \boldsymbol{x}^{(k)}, \boldsymbol{z}(\boldsymbol{x}^{(k)})) = \boldsymbol{0}$. If this is not the case the policy for the next iteration is generated in step 4. The function iteration algorithm in 4(a) determines it as a mixture of the last and the expectational solution. In the example model damping was not necessary and I took $\alpha = 0$ for the fastest convergence so that the expectational solution is taken as the policy for the next iteration. In case of the Newton algorithm in 4(b) the next policy is calculated according to the Jacobian of the residuals. Both methods differ only in step 4 where a policy for the next iteration is calculated. They repeat steps 1.-6. until a sufficiently accurate solution with (weighted) residuals close to zero is found.
The iteration is described for a Kronecker product approximation in the approximation step 1 and the interpolation step 2(a)ii. The Smolyak operator is a linear combination of the Kronecker operator and the procedure is therefore analogous. A finite elements approximation strategy needs a different step 2(a)ii where the interpolation of the next period policy is calculated. Moreover the state grid is different since for finite elements the grid can be chosen freely. If the Galerkin weighted residuals are used it changes step 1 where the approximation parameters are determined.
For a large grid the Jacobian is costly to calculate and I use Broyden's variant of the Newton algorithm. The inverse of the Jacobian $[\partial\, r(\boldsymbol{s}, \boldsymbol{x}^{(k)})/\partial\, \boldsymbol{x}]^{-1}$ is approximated by a matrix $\boldsymbol{A}^{(k)}$ and updated according to the residual starting with an initial identity matrix.
The trade off between the function iteration and the Broyden algorithm is that the Jacobian based steps converge faster but the function iteration is more robust for any start values. I implemented both methods. The function iteration is used while searching for the modes of the posterior density with the solution of the linearized model as starting values for the policies. In subsequent likelihood evaluations when sampling around the posterior mode the structural parameters and policies do not change much. Then it is save to use Broyden's method with the nonlinear policy at the previous structural parameters as starting values. This implies that calculating the solution takes only a small fraction of the complete running time of the estimation process. The most time consuming part is the evaluation of the likelihood for a given nonlinear policy. This may change for

larger models.

3.8 Efficient Calculation

In the estimation procedure some speed reduction ideas can be exploited. For repeated likelihood evaluations the model has to be solved very often. Some parts of the integration and approximation algorithms can be factored out and done only once.
The first saving is rather simple and obvious and allows to compute Gauss-Hermite nodes and weights only once. In each iteration over the structural parameters within the estimation process, the covariance matrix of the involved shocks changes and implies different nodes for the approximation of the rational expectation integral. This is also true for the evaluation of the likelihood where moments of nonlinear transformations of normal random variables with changing moments have to be calculated. Since the Gauss-Hermite formulas deliver nodes and weights for a standard normal density function, changing moments can be taken into account by a simple change of variables. Nodes $\boldsymbol{y}^{(i)}$ for the standard normal density can be transformed by $\boldsymbol{x}^{(i)} = \boldsymbol{\mu} + \boldsymbol{y}^{(i)}\boldsymbol{\Sigma}^{1/2}$ to obtain nodes for the general density $\mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma})$ of $\boldsymbol{x}$, where $\boldsymbol{\Sigma}^{1/2}$ is the Cholesky factor of $\boldsymbol{\Sigma}$.
Possible acceleration schemes for the policy function approximation are more involved. Before I present a method for efficiency gains in the matrix variant of approximation in equation (15), I want to discuss a method which takes advantage of the known closed form expressions for the Chebyshev polynomial coefficients. A univariate ($d = 1$) approximation is defined by

$$f(x) \approx U^i(f) = \sum_{j=1}^{m_i} c_j^i \, T_{j-1}(x)$$

where m_i is the Clenshaw-Curtis formula $m_i = 2^{i-1} + 1, m_1 = 1$ and $T_{j-1}(x)$ is a Chebyshev polynomial of degree $j-1$. In case of Chebyshev roots as nodes $x_{i,k} = \cos\left(\frac{\pi(k-0.5)}{m_i}\right)$ the closed form expression for the coefficients is

$$c_j^i = \frac{\sum_{k=1}^{m_i} f(x_{i,k}) T_{j-1}(x_{i,k})}{\sum_{k=1}^{m_i} (T_{j-1}(x_{i,k}))^2}$$

and in case of nested Gauss-Lobatto nodes $x_{i,k} = -\cos(\frac{\pi(k-1)}{m_i - 1})$ we have

$$c_j^i = \frac{d_{i,j}}{\max(1/2, m_i - 1)} \sum_{k=1}^{m_i} \frac{1}{3 - d_{i,k}} f(x_{i,k}) T_{j-1}(x_{i,k})$$

where $d_{i,j} = 1$ for $j = 1, m_i$ and $d_{i,j} = 2$ else. Chebyshev polynomials have a trigonometric representation $T_j(x) = \cos(j \cos^{-1} x)$ and a fast way to calculate

the coefficients is the discrete fast Fourier transform. For the usually real valued functions in economics, further simplification applies the real form of the transform, called cosine transform, as described for example in Press, Flannery, Teukolsky, and Vetterling (1988).
The multivariate ($d > 1$) extension with $\mathbf{x} = [x_1 \ \dots \ x_d]$ can be written as

$$f(\mathbf{x}) \approx (V^{i_1} \otimes \cdots \otimes V^{i_d})(f) = \sum_{j_1=1}^{m_{i_1}} \cdots \sum_{j_d=1}^{m_{i_d}} c_{j_1,\dots,j_d}^{i_1,\dots,i_d} T_{j_1-1}(x_1) \cdots T_{j_d-1}(x_d).$$

In case of Chebyshev roots as nodes the coefficients can be calculated by

$$c_{j_1,\dots,j_d}^{i_1,\dots,i_d} = \frac{\sum_{k_1=1}^{m_{i_1}} \cdots \sum_{k_d=1}^{m_{i_d}} f(\{x_{i_1,k_1}, \dots, x_{i_d,k_d}\}) T_{j_1-1}(x_{i_1,k_1}) \cdots T_{j_d-1}(x_{i_d,k_d})}{(\sum_{k_1=1}^{m_{i_1}} (T_{j_1-1}(x_{i_1,k_1}))^2) \cdots (\sum_{k_d=1}^{m_{i_d}} (T_{j_d-1}(x_{i_d,k_d}))^2)}$$

whereas the Gauss-Lobatto grid results in

$$\begin{aligned} c_{j_1,\dots,j_d}^{i_1,\dots,i_d} &= \frac{d_{i_1,j_1} \cdots d_{i_d,j_d}}{\max(1/2, m_{i_1} - 1) \cdots \max(1/2, m_{i_d} - 1)} \\ &\quad \sum_{k_1=1}^{m_{i_1}} \cdots \sum_{k_d=1}^{m_{i_d}} \frac{f(\{x_{i_1,k_1}, \dots, x_{i_d,k_d}\}) T_{j_1-1}(x_{i_1,k_1}) \cdots T_{j_d-1}(x_{i_d,k_d})}{(3 - d_{i_1,k_1}) \cdots (3 - d_{i_d,k_d})}. \end{aligned}$$

The coefficients of a multivariate approximation can be calculated by repeated application of the univariate discrete Fourier transform. Specialized multivariate transform algorithms drop the calculation costs further to a fraction of around $2/d$.
It is clear that the closed form expressions and the fast Fourier transform replicate the matrix inversion $\mathbf{c} = \mathbf{T}^{-1}(\boldsymbol{s}) f(\boldsymbol{s})$ in equation (15). The insight I used is that the inverted Chebyshev matrix $\mathbf{T}^{-1}(\boldsymbol{s})$ in step 1 in table 2 is the same for each iteration when solving for the policy function. It is even the same for all likelihood evaluations if one restricts the space of approximation to be the same for all iterations over structural parameters. Therefore the closed form inversion can be outperformed by a large factor if the Chebyshev basis matrices are inverted only once. The calculation of coefficients in each root finding iteration reduces thereby to a simple matrix multiplication $\mathbf{c} = \mathbf{T}^{-1}(\boldsymbol{s}) f(\boldsymbol{s})$. This is even faster than the repeated fast Fourier transform equivalent.
A simplification of the matrix inversion can be achieved if one uses the Kronecker inversion theorem $\boldsymbol{T}^{-1} = (\boldsymbol{T}_1 \otimes \cdots \otimes \boldsymbol{T}_d)^{-1} = \boldsymbol{T}_1^{-1} \otimes \cdots \otimes \boldsymbol{T}_d^{-1}$. This inversion appears in the approximation process for the Kronecker product extension (and therefore in the Smolyak operator) since the multivariate basis matrix $\boldsymbol{T}$ is the Kronecker product of the univariate basis matrices $\boldsymbol{T}_i$. But this cost reduction is of minor importance since the inversion has to be done only once.
The reduction scheme in the matrix variant does not take into account the special structure of the Smolyak operator. It reduces the costs of a product rule

approximation and carries over to the Smolyak operator where product rules are combined linearly. The coefficients $c_{j_1,\ldots,j_d}^{i_1,\ldots,i_d}$ are in fact products of the underlying univariate coefficients. Together with the hierarchical structure of the Smolyak operator and its linear Kronecker product combinations there are many repeated calculations. These efficiency gains are exploited by Petras (1999) who carefully remembers which underlying basic coefficients were already calculated when processing through the Smolyak levels. Again this is a matrix inversion equivalent and it is not clear whether it is faster compared to the simple multiplication with a once and for all inverted basis matrix.
Even after all tricks to lower running times one computer may be too slow for an interesting model. This can be expected to happen in case of heterogeneous agents models like OLG models or international macroeconomic models when several countries with possibly several sectors within each country have to be modelled. A recent development in the computer industry is to connect several standard desktop computers to form a cluster and divide the problem into subproblems which can then be solved by the connected computers in parallel processes. Afterwards the partial results are combined to the overall result. A prerequisite for such a procedure is that the problem can be divided in such a way that the communication needed between the involved computer does not exceed the time advantage of dividing the problem. The methods used should therefore be suitable for parallelization. This is the case for the finite element approximation since the approximation space is effectively divided into subspaces and can therefore be distributed to single computers. For spectral approximation dividing the solution process is more difficult if not impossible. However, the proposed Metropolis-Hastings algorithm allows to parallelize the estimation process even if the approximation cannot be distributed.

3.9 Euler Error

The approximation procedure should be accompanied by error estimates to assure convergence and to control the approximation quality. The exact policy functions would give zero first order condition residuals for all states. Each deviation is therefore due to the approximation once the policy function is substituted for the policy variable.
What says a certain number of the residual about the approximation quality? Judd (1992) proposed to normalize the residual which can then be interpreted in an economically meaningful way. This can be done by dividing the residuals of the Euler equation by the marginal utility. For the interpretation it is helpful to do an intermediate step in the model at hand and isolate consumption. Dividing the residual by $(1-x_l(\boldsymbol{s_t}))^{(1-\theta)(1-\tau)}$ and taking it to the power of $1/(\theta(1-\tau)-1)$

gives the Euler equation in terms of consumption

$$\begin{aligned} r^c(\boldsymbol{s}) &= x^c(\boldsymbol{s}) - \left(\frac{\beta \mathrm{E}_t h(\boldsymbol{s}, \boldsymbol{x}^*(\boldsymbol{s}), \boldsymbol{e}, \boldsymbol{s}', \boldsymbol{x}^*(\boldsymbol{s}'))}{(1 - x^l(\boldsymbol{s}))^{(1-\theta)(1-\tau)}} \right)^{\frac{1}{\theta(1-\tau)-1}} \\ \boldsymbol{s}' &= \boldsymbol{g}(\boldsymbol{s}, \boldsymbol{x}^*(\boldsymbol{s}), \boldsymbol{e}'). \end{aligned}$$

Dividing this expression by consumption and taking its absolute value

$$r^E = \left| \frac{r^c(\boldsymbol{s})}{\psi_c(\boldsymbol{s})} \right|$$

we obtain the Euler error for a given state vector and a nice interpretation. The Euler error r^E is the fraction of consumption expenditures lost by relying on an approximated rather then exact policy. The logarithm of the error $\log_{10} r^E$ delivers a more informative plot. A log Euler error of -3 says that one has to consume 1000 units before one is lost. The maximal error in the whole approximation space can finally be taken to characterize the approximation quality.

4 Likelihood

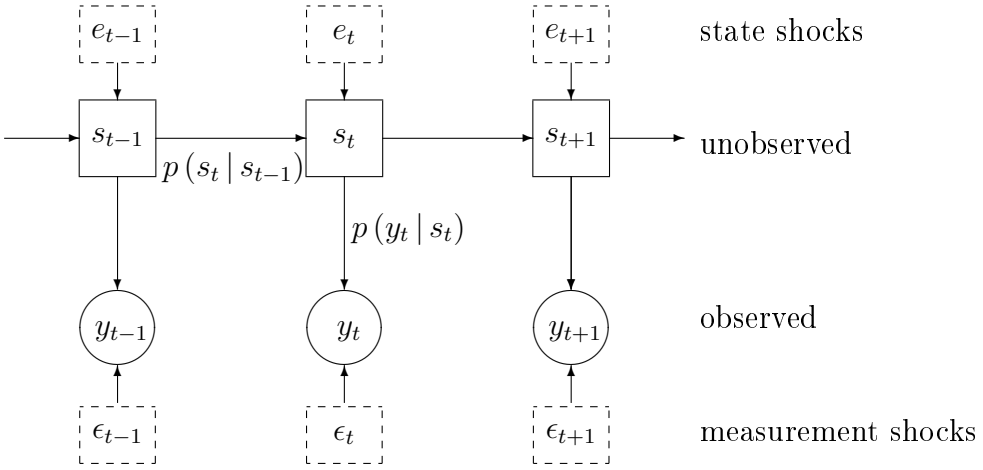
Once the solution of the model $\boldsymbol{x}_t = \boldsymbol{x}^*(\boldsymbol{s}_t)$ is obtained, the implied dynamics can be compared to data. The main problem on the way to analyze the density of observables of the state space model is to derive the density of unobservables. The observables density evaluated at the realizations $\boldsymbol{y}^0$ is functionally equivalent to the likelihood. For a linear state space model with Gaussian shocks the Kalman (1960) filter provides a closed form solution for both densities. Nonlinearity and non-Gaussian distributed state and measurement shocks make it in general impossible to arrive at analytical expressions and approximations are needed.
The state space model is described by the state and measurement equations

$$\boldsymbol{s}_{t+1} = \boldsymbol{g}(\boldsymbol{s}_t, \boldsymbol{x}^*(\boldsymbol{s}_t), \boldsymbol{e}_{t+1}) = \boldsymbol{g}^*(\boldsymbol{s}_t, \boldsymbol{e}_{t+1}) \tag{25}$$

$$\boldsymbol{y}_t = \boldsymbol{m}(\boldsymbol{s}_t, \boldsymbol{\epsilon}_t) \tag{26}$$

with unobservables $\boldsymbol{s}_t$ and observables $\boldsymbol{y}_t$. State $\boldsymbol{e}_t$ and measurement shocks $\boldsymbol{\epsilon}_t$ follow some distribution. The structure of the state space model is summarized in the Bayes net in figure 6. If the the policy function is plugged into the state equation it implies the equilibrium transition $\boldsymbol{g}^*$. In the following I will drop the starred notation and the state transition equation with two arguments is implicitly assumed to be the equilibrium transition. In general the model equations can be time dependent, for example through varying variances or other changing parameters.
The autoregressive formulation in equations (25) and (26) is convenient for the numerical and economic analysis. It describes a nonlinear transformation of the

Figure 6: State Space Model



involved random variables and results in some nonstandard densities for the observables and unobservables. For the statistical discussion it is useful to alternatively describe the model by three densities

$$p(\boldsymbol{y}_t \mid \boldsymbol{s}_t) \qquad p(\boldsymbol{s}_t \mid \boldsymbol{s}_{t-1}) \qquad \text{for } t = 1, ..., T \qquad p(\boldsymbol{s}_0).$$

The first density represents the measurement equation and relates the unobserved to the observed variables. The second density describes the equilibrium state transition in time. The third is the initial information about the state before data is observed. The unobserved state process is Markovian with $p(\boldsymbol{s}_t \mid \boldsymbol{s}_{1:t-1}) = p(\boldsymbol{s}_t \mid \boldsymbol{s}_{t-1})$ and observations are conditional independent given state $p(\boldsymbol{s}_t \mid \boldsymbol{s}_{t-1}, \boldsymbol{y}_{1:t-1}) = p(\boldsymbol{s}_t \mid \boldsymbol{s}_{t-1})$. The notation $\boldsymbol{s}_{1:t}$ is used for a sequence of vectors $\{\boldsymbol{s}_1, ..., \boldsymbol{s}_t\}$. All model related densities are without being noted conditioned on model M_i and the associated structural parameters $\boldsymbol{\theta}_{M_i}$. The full notation will be needed for the model selection criterion in section 6.
The state space model is much more general than the various ARMA models. It is built upon the distinction between observed and unobserved variables. This distinction characterizes most economic theories and many econometric complications and offers the advantage of a unified framework. Stochastic trends can be easily modelled so that nonstationary data can be decomposed in an economically sensible way. It offers the statistical model for a unified growth and business cycle literature. Moreover it allows for cointegration relations, missing observations, measurement errors, learning processes or time varying coefficients as in regime switching models as well as time varying shock distributions as in GARCH models.
In engineering applications the density of the unobserved states is often the end

of an investigation. The parameters and the functional form of the model are known and represent for example physical laws. The evaluation of these laws in the light of data is not the purpose of the analysis. In the original application of the state space model the unobserved state was the position of a satellite inferred from noisy measurement of a known transition law in the orbit. The econometric applications of unobserved state estimation are for example the Hodrick-Prescott decomposition of the output in a trend and business cycle component. They are not observable separately but have to be inferred from one single output time series. In its usual formulation the HP filter gives an estimate of the unobserved trend as the solution to the minimization problem

$$\min_{y_t^*} \sum_{t=1}^{T} \frac{1}{\sigma_0^2}(y_t - y_t^*)^2 + \frac{1}{\sigma_1^2}(\Delta^2 y_t^*)^2$$

where y is the observed output, y^* is the unobserved trend, σ_0^2 is the variance of the business cycle $y - y^*$ and σ_1^2 is the variance of the growth rate of the trend. The minimization is invariant to a monotone transformation and what matters is $\lambda = \sigma_0^2/\sigma_1^2$. This parameter depends on the frequency of the data and is usually chosen to be 100 for annual, 400 for semi-annual and 1600 for quarterly data. Harvey (1985) shows the state space representation of the HP filter. The measurement equation is the sum of the unobserved trend and business cycle fluctuation

$$y_t = y_t^* + \epsilon_t$$

with $\epsilon_t \sim \mathcal{N}(0, \sigma_0^2)$. The state equations define the growth rate of the trend

$$\begin{aligned} y_t^* &= g_{t-1} + y_{t-1}^* \\ g_t &= g_{t-1} + e_t \end{aligned}$$

with $e_t \sim \mathcal{N}(0, \sigma_0^2/\lambda)$. The advantage of this representation is that it uncovers the explicit assumption for the trend process. In the HP filter the change of the trend follows a random walk. The Kalman filter provides the unobserved trend $y_{1:T}^*$ as the smoothed estimate of the unobservable state. It can then be used to recover the business cycle as the residual in the measurement equation.
Most often the economic focus is on parameter estimates. In the engineering literature the parameter estimation is called joint (state and parameter) estimation or simply the static problem. The static classification is due to the fact that parameters and states are both treated as unobservable random variables and are therefore statistically not fundamentally different. They differ in the fact that parameters are fixed or static whereas unobserved states follow a dynamic process. The most simple approach to parameter estimation is to augment the state vector by the vector of parameters and to transform the parameter estimation into a problem of the state estimation. It is not clear whether this works analogously

for economics where we are interested in the structural and not the reduced form parameters. Doucet, de Freitas, and Gordon (2001) offer an extensive overview of nonlinear filters.
The problems of likelihood evaluation, prediction or unobserved state estimation have to be solved by deriving the density of the unobserved states. I use the filtering algorithms to derive the likelihood of the sample for a given parameter vector.

4.1 State Space

The state space form of the economic model is obtained once the policy functions for labor and consumption $x^l(k,a)$ and $x^c(k,a)$ are calculated. In the example model the state equations describe the capital and productivity dynamics

$$\begin{aligned} k_t &= e^{a_{t-1}} k_{t-1}^{\alpha} x^l(k_{t-1}, a_{t-1})^{1-\alpha} - x^c(k_{t-1}, a_{t-1}) + (1-\delta)k_{t-1} \\ a_t &= \rho a_{t-1} + e_t. \end{aligned}$$

The measurement equations for output y_t, labor l_t and investment i_t are

$$\begin{aligned} y_t &= e^{a_t} k_t^{\alpha} x^l(k_t, a_t)^{1-\alpha} + \epsilon_t^y \\ l_t &= x^l(k_t, a_t) + \epsilon_t^l \\ i_t &= e^{a_t} k_t^{\alpha} x^l(k_t, a_t)^{1-\alpha} - x^c(k_t, a_t) + \epsilon_t^i. \end{aligned}$$

These measurement equations do not have the general but an additive noise form with $\boldsymbol{m}(\boldsymbol{s}_t, \boldsymbol{\epsilon}_t) = \boldsymbol{m'}(\boldsymbol{s}_t) + \boldsymbol{\epsilon}_t$. This feature simplifies the derivation of the period contributions to the likelihood. In this model formulation it is necessary to add measurement errors because there is only one driving structural shock but three observables. Without these additional shocks there would be a deterministic functional dependency between the observed variables. An economically more sensible way to solve this problem is to add structural state shocks or to use less observables. Since the economic analysis is not of primary interest in this work I simply add the measurement errors as in Fernádez-Villaverde and Rubio-Ramírez (2004b). Ruge-Murcia (2003) discusses this topic more extensively.

4.2 Filtering

The estimation approach is recursive and uses the Markovian conditional independence of the state equation. Each new observation is used to update the information about the unobservables summarized by the posterior density of the unobserved states. For satellite maneuvers for example this recursive structure corresponds to the real situation since noisy measurements of the satellite's position become available during the control process.

The main tool for the filtering problem is again Bayes' formula. Here it describes how the data helps to learn about the unobservables. The prior density of unobservables describes the available information before the new data has been analyzed. The evidence about the observables incorporated in the likelihood informs us by transforming the prior into the posterior. Once the data is processed the posterior becomes the prior with regard to the new data. Therefore the concept of prior and posterior is defined relative to the new data.
The filtering recursion is a two step procedure. The first step is to form a prediction and the second is the filtering step where new information from data is incorporated to modify the prediction. The state vector contains all information about the system and we start with prior information

$$p(\boldsymbol{s}_0) = p(\boldsymbol{s}_0 \mid \boldsymbol{y}_0).$$

The prediction step represents the prior density $p(\boldsymbol{s}_t \mid \boldsymbol{y}_{1:t-1})$

$$\begin{aligned} p(\boldsymbol{s}_t \mid \boldsymbol{y}_{1:t-1}) &= \int p(\boldsymbol{s}_t, \boldsymbol{s}_{t-1} \mid \boldsymbol{y}_{1:t-1})\, d\boldsymbol{s}_{t-1} && (27) \\ &= \int p(\boldsymbol{s}_t \mid \boldsymbol{s}_{t-1})\, p(\boldsymbol{s}_{t-1} \mid \boldsymbol{y}_{1:t-1})\, d\boldsymbol{s}_{t-1}. && (28) \end{aligned}$$

It is formed by the weighted state transition $p(\boldsymbol{s}_t \mid \boldsymbol{s}_{t-1})$ with the weight given by the last period posterior $p(\boldsymbol{s}_{t-1} \mid \boldsymbol{y}_{1:t-1})$. The next posterior $p(\boldsymbol{s}_t \mid \boldsymbol{y}_{1:t})$ is obtained in the filtering step and updates the prediction by incorporating new data $\boldsymbol{y}_t$

$$p(\boldsymbol{s}_t \mid \boldsymbol{y}_{1:t}) = \frac{p(\boldsymbol{s}_t, \boldsymbol{y}_t \mid \boldsymbol{y}_{1:t-1})}{p(\boldsymbol{y}_t \mid \boldsymbol{y}_{1:t-1})} = l_t^{-1} p(\boldsymbol{y}_t \mid \boldsymbol{s}_t)\, p(\boldsymbol{s}_t \mid \boldsymbol{y}_{1:t-1}). \tag{29}$$

Therefore these two recursion steps transform one posterior $p(\boldsymbol{s}_{t-1} \mid \boldsymbol{y}_{1:t-1})$ into the next $p(\boldsymbol{s}_t \mid \boldsymbol{y}_{1:t})$. The filtering equation (29) is the result of a repeated application of Bayes' formula.[3] The normalizing constant l_t in the filtering step

$$l_t = \int p(\boldsymbol{y}_t \mid \boldsymbol{s}_t)\, p(\boldsymbol{s}_t \mid \boldsymbol{y}_{1:t-1})\, d\boldsymbol{s}_t = p(\boldsymbol{y}_t \mid \boldsymbol{y}_{1:t-1}) \tag{30}$$

is the period contribution to the likelihood

$$\mathcal{L}(\boldsymbol{\theta}; \boldsymbol{y}_{1:T}) \equiv p(\boldsymbol{y}_{1:T} \mid \boldsymbol{\theta}) = \prod_{t=1}^{T} p(\boldsymbol{y}_t \mid \boldsymbol{y}_{1:t-1}, \boldsymbol{\theta}) = \prod_{t=1}^{T} l_t.$$

---

[3] $p(\boldsymbol{s}_t \mid \boldsymbol{y}_{1:t}) \overset{(1)}{=} \frac{p(\boldsymbol{y}_{1:t} \mid \boldsymbol{s}_t)\, p(\boldsymbol{s}_t)}{p(\boldsymbol{y}_{1:t})} \overset{(2)}{=} \frac{p(\boldsymbol{y}_t, \boldsymbol{y}_{1:t-1} \mid \boldsymbol{s}_t)\, p(\boldsymbol{s}_t)}{p(\boldsymbol{y}_t, \boldsymbol{y}_{1:t-1})} \overset{(3)}{=} \frac{p(\boldsymbol{y}_t \mid \boldsymbol{y}_{1:t-1}, \boldsymbol{s}_t)\, p(\boldsymbol{y}_{1:t-1} \mid \boldsymbol{s}_t)\, p(\boldsymbol{s}_t)}{p(\boldsymbol{y}_t \mid \boldsymbol{y}_{1:t-1})\, p(\boldsymbol{y}_{1:t-1}))} \overset{(4)}{=} \frac{p(\boldsymbol{y}_t \mid \boldsymbol{y}_{1:t-1}, \boldsymbol{s}_t)\, p(\boldsymbol{s}_t \mid \boldsymbol{y}_{1:t-1})\, p(\boldsymbol{y}_{1:t-1})\, p(\boldsymbol{s}_t)}{p(\boldsymbol{y}_t \mid \boldsymbol{y}_{1:t-1})\, p(\boldsymbol{y}_{1:t-1})\, p(\boldsymbol{s}_t)} \overset{(5)}{=} \frac{p(\boldsymbol{y}_t \mid \boldsymbol{s}_t)\, p(\boldsymbol{s}_t \mid \boldsymbol{y}_{1:t-1})}{p(\boldsymbol{y}_t \mid \boldsymbol{y}_{1:t-1})} \overset{(6)}{=} \frac{p(\boldsymbol{y}_t \mid \boldsymbol{s}_t)\, p(\boldsymbol{s}_t \mid \boldsymbol{y}_{1:t-1})}{\int p(\boldsymbol{y}_t \mid \boldsymbol{s}_t)\, p(\boldsymbol{s}_t \mid \boldsymbol{y}_{1:t-1})\, d\boldsymbol{s}_t}$
where: (1) Bayes' formula, (2) separate: $\boldsymbol{y}_{1:t} = \{\boldsymbol{y}_t, \boldsymbol{y}_{1:t-1}\}$, (3) factorize: $\frac{p(a,b \mid c) = p(a \mid b,c)\, p(b \mid c)}{p(a,b) = p(a \mid b)\, p(b)}$, (4) Bayes' formula, (5) cancel terms: $p(\boldsymbol{y}_{1:t-1})\, p(\boldsymbol{s}_t)$, use conditional independence: $p(\boldsymbol{y}_t \mid \boldsymbol{y}_{1:t-1}, \boldsymbol{s}_t) = p(\boldsymbol{y}_t \mid \boldsymbol{s}_t)$

of the complete sample. The state posterior densities $p(\boldsymbol{s}_t \,|\, \boldsymbol{y}_{1:t})$ for $t = 1, ..., T$ are either an end in itself as in the HP filter or an instrument to obtain the sample likelihood $\mathcal{L}$. The period contribution to the likelihood in the last equation and in (30) is implicitly conditioned on the parameter vector $\boldsymbol{\theta}$. This is without being further noted also true for all model related densities in the rest of this section. The likelihood can be either maximized over the parameter vector or as in this work used to estimate the parameter posterior density $p(\boldsymbol{\theta} \,|\, \boldsymbol{y}_{1:T})$. This will be discussed in section 5.
If we are interested in the estimates of the unobserved states then the posterior density estimates can be further refined. This so called smoothing procedure works recursively backwards in time for $t = T, ..., 1$ where the state densities $p(\boldsymbol{s}_t \,|\, \boldsymbol{y}_{1:T})$ conditioned on the complete sample are derived. Smoothing algorithms are not discussed since the focus is to evaluate the likelihood.
For the likelihood evaluation at least the first two moments of the posterior density of the unobserved states have to be estimated. This involves repeated integration of a density weighted nonlinear function. That means we need to approximate the expected value of a nonlinearly transformed random variable

$$\mathcal{I}(f)_p = E(f(\boldsymbol{s})) = \int f(\boldsymbol{s})\, p(\boldsymbol{s})\, d\boldsymbol{s}. \tag{31}$$

The methods used for these integrals are either a Monte Carlo simulation or Gaussian quadrature as discussed in section 3.4. In the filtering literature another so called unscented transform is often used. It is a numerical integration technique developed for nonlinear filtering by Julier and Uhlmann (1997), extended in Julier and Uhlmann (2002) and summarized in Julier and Uhlmann (2004). It is a deterministic integration strategy like Gaussian quadrature.
The unscented filter tries to cope with the shortcomings of the traditional extended Kalman filter for nonlinear state space models. The extended Kalman filter uses a first order Taylor approximation of the nonlinear state space equations in order to approximate the first two moments of the nonlinear transformation of a Gaussian random variable. Since these transformed moments can be given in a closed form for the linearized equations, the hope is that they are a reasonable approximation of the true moments of the nonlinear transform. This is in general not true.
The unscented transform constructs nodes and weights in order to approximate the mean and the covariance of a nonlinear transformation instead of approximating the state transition function. The nonlinear function is evaluated at the nodes and a weighted sum is the integral approximation. The derivation of these nodes relies on identifying conditions which match the true moments of a nonlinear transform. Of course the general idea is the same as in Gaussian quadrature, namely that a density is easier to approximate than a general function. In fact the unscented transform uses low level Gaussian quadrature nodes. The exponentially rising computational burden of the Kronecker product quadrature formulas

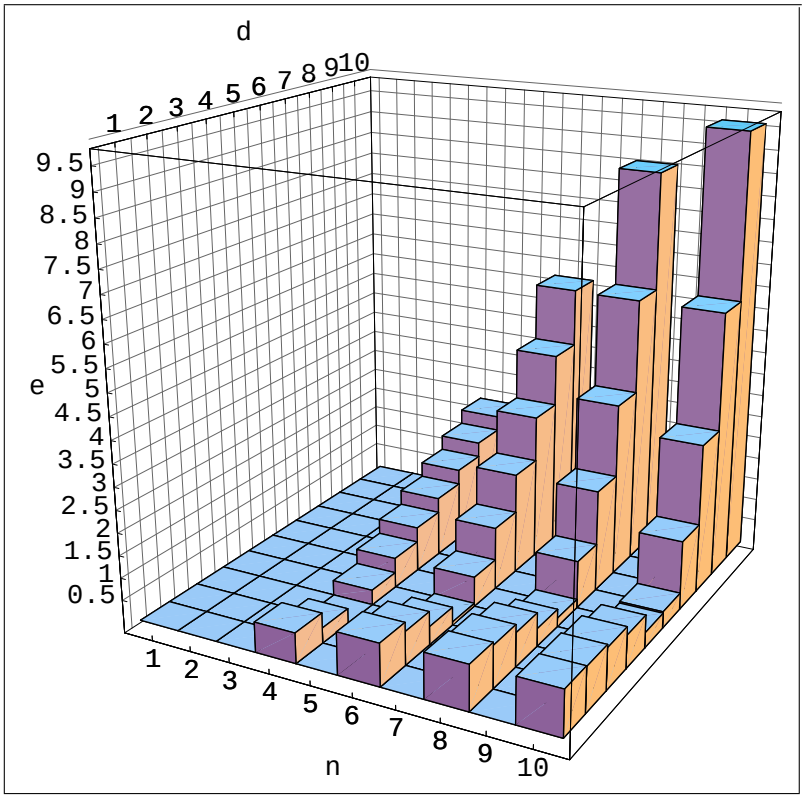
is the usual argument in favor of the unscented transform where the number of nodes is restricted to $2d+1$ and d is the dimension of the random variable.
This restriction of the number of nodes may be a problem in economic models since the policy functions we usually need are of higher polynomial degree. These policy functions are one part of the state equations which have to be integrated in the filtering recursions.
How much decreases the approximation accuracy of the unscented transform with an increasing dimension and polynomial degree of the nonlinear function? If it does not decrease there would be no curse of dimensionality and costs would rise linearly as in $2d+1$. The unscented transform is not a solution of the curse in any respect but merely a shift of the problem from the rising number of nodes for a given accuracy to a given number of nodes with a decreasing accuracy.
In order to gain some intuition for the errors that may occur I calculated an example for a standard normally distributed random variable. I use the unscented nodes and weights to approximate the expected value of a nonlinear transformation of a random variable while increasing the dimension. In order to check the unscented transform's ability to handle different degrees of nonlinearity I take a simple polynomial as integrand. It represents functions with different degree of nonlinearity. The d-dimensional polynomial of degree n is $f_{d,n}(\boldsymbol{x}) = \sum_{i=1}^{d} x_i^n$, where x_i is the i^{th} element of vector $\boldsymbol{x}$. It is integrated over a standard normal uncorrelated random variable with $p(\boldsymbol{x}) = \mathcal{N}(\boldsymbol{x}; \hat{\boldsymbol{x}}, \boldsymbol{\Sigma}) = \mathcal{N}(\boldsymbol{x}; \boldsymbol{0}, \boldsymbol{I})$.[4] Since the function is additive and the variables are uncorrelated the true expected value is $\mathcal{I}(f_{d,n})_p = dr^n$ where r^n is the n^{th} raw moment of a standard normally distributed random variable. The first 10 moments are 0, 1, 0, 3, 0, 15, 0, 105, 0, 945. According to Julier and Uhlmann (2004) the unscented nodes for the expected value approximation are given by

$$\begin{aligned} \boldsymbol{x}_j &= \hat{\boldsymbol{x}} + \sqrt{d\boldsymbol{\Sigma}_j} \qquad \text{for } j = 1, ..., d \\ \boldsymbol{x}_j &= \hat{\boldsymbol{x}} - \sqrt{d\boldsymbol{\Sigma}_{j-d}} \qquad \text{for } j = d+1, ..., 2d \end{aligned}$$

with equal weights $w_j = 1/(2d)$. $\boldsymbol{\Sigma}_j$ is the j^{th} column of the Cholesky matrix square root. For a standard normal distribution $\boldsymbol{\Sigma}$ is the identity matrix and the approximation is the simple mean of the function evaluations $\hat{\mathcal{I}}(f_{d,n})_p = \sum_{j=1}^{2d} f(\boldsymbol{x}_j)/(2d)$. The nonlinear function is evaluated at $2d$ vectors with all elements being zero except one which is once $+\sqrt{d}$ and once $-\sqrt{d}$. The approximation of the nonlinear transformation is the mean of the function at these vectors. For the approximation of the covariance the nodes and weights are slightly different. They are parameterized and can capture some non normal higher moments. Since the nonlinear function is symmetric in each variable each evaluation gives $\sqrt{d}^n$. This is also the mean and therefore the approximation is $\hat{\mathcal{I}}(f_{d,n}) = \sqrt{d}^n$ for even n and zero for uneven n. Since the uneven moments of a normal density

[4] The notation of a n-variate normal density is $\mathcal{N}(\boldsymbol{s}; \boldsymbol{\mu}, \boldsymbol{\Sigma}) \equiv ((2\pi)^{d_s} \mid \boldsymbol{\Sigma} \mid)^{-1/2} \exp\left(-[\boldsymbol{s}-\boldsymbol{\mu}]^T \boldsymbol{\Sigma}^{-1} [\boldsymbol{s}-\boldsymbol{\mu}]/2\right)$.

Figure 7: Relative Error of the Unscented Transform



are zero and the nodes are symmetric and equally weighted, the approximation will be exact for uneven polynomial degrees. The percentage error of the approximation depends on the dimension and the polynomial degree of the nonlinear function f and is given by $e(d,n) = (\hat{\mathcal{I}} - \mathcal{I})/\mathcal{I}$ for even d and zero for uneven n. Figure 7 plots this function. It shows that for a polynomial degree of $n = 2$ the unscented approximation error is zero for any dimension, i.e. the unscented integration has a polynomial exactness of second degree. If a higher polynomial exactness is needed the error rises immediately to .33 and even more for higher dimensions. This approximation strategy will therefore run into problems where polynomial exactness beyond the second degree is needed. The curse of dimensionality is in the unscented transform not an increasing computational burden for a given polynomial exactness but a decreasing accuracy due to an insufficient number of nodes and weights in higher dimensions.

A Smolyak quadrature based filter can be expected to perform better over a wider class of models. It allows a true relieve of the curse of dimensionality with a flexible accuracy according to the needs for a given model. Moreover a Smolyak based Gaussian quadrature filter has the advantage of a more accurate approximation of

non-normal densities whereas in the unscented transform skewness and kurtosis are parameterized in an obscure way to account for non-normal densities.
My focus in the following nonlinear filter discussion is on the question where integration arises and Monte Carlo methods can be substituted by Smolyak based Gaussian quadrature.

4.3 Kalman Filter

This section derives the Kalman gain as an exact solution for the linear Gaussian model and as an approximation for nonlinear models. The approximation will be used for the Gaussian (quadrature) filters in later sections.

4.3.1 Nonlinear

The nonlinear approximation starts with the decomposition of the posterior density by the Bayes formula

$$p\left(\boldsymbol{s}_t \,|\, \boldsymbol{y}_{1:t}\right) = \frac{p\left(\boldsymbol{s}_t, \boldsymbol{y}_{1:t}\right)}{p\left(\boldsymbol{y}_{1:t}\right)}.$$

For normalized densities we can write $p\left(\boldsymbol{s}_t \,|\, \boldsymbol{y}_{1:t}\right) = p\left(\boldsymbol{s}_t, \boldsymbol{y}_{1:t}\right)$. If we approximate the joint density by the predictive density $p\left(\boldsymbol{s}_t, \boldsymbol{y}_{1:t}\right) \approx p\left(\boldsymbol{s}_t, \boldsymbol{y}_t \,|\, \boldsymbol{s}_{t-1}, \boldsymbol{y}_{1:t-1}\right)$ the Gaussian posterior is given by

$$\begin{aligned} p\left(\boldsymbol{s}_t \,|\, \boldsymbol{y}_{1:t}\right) &= \mathcal{N}(\hat{\boldsymbol{s}}, \boldsymbol{P}^{\hat{\boldsymbol{s}}\hat{\boldsymbol{s}}}) \\ &= \frac{1}{(2\pi)^{d_x/2} \,|\, \boldsymbol{P}^{\hat{\boldsymbol{s}}\hat{\boldsymbol{s}}} \,|^{\,1/2}} \exp(-\frac{1}{2}A) \end{aligned}$$

with

$$\begin{aligned} \hat{\boldsymbol{s}} &= \hat{\boldsymbol{s}}_{t|t} = E(\boldsymbol{s}_t \,|\, \boldsymbol{y}_{1:t}) \\ \boldsymbol{P}^{\hat{\boldsymbol{s}}\hat{\boldsymbol{s}}} &= \boldsymbol{P}^{ss}_{t|t} = E([\boldsymbol{s}_t - \hat{\boldsymbol{s}}][\boldsymbol{s}_t - \hat{\boldsymbol{s}}]^T \,|\, \boldsymbol{y}_{1:t}) \\ A &= [\boldsymbol{s}_t - \hat{\boldsymbol{s}}]^T (\boldsymbol{P}^{\hat{\boldsymbol{s}}\hat{\boldsymbol{s}}})^{-1} [\boldsymbol{s}_t - \hat{\boldsymbol{s}}] \\ &= \boldsymbol{s}_t^T (\boldsymbol{P}^{\hat{\boldsymbol{s}}\hat{\boldsymbol{s}}})^{-1} \boldsymbol{s}_t - \boldsymbol{s}_t^T (\boldsymbol{P}^{\hat{\boldsymbol{s}}\hat{\boldsymbol{s}}})^{-1} \hat{\boldsymbol{s}} \\ &\quad -\hat{\boldsymbol{s}}^T (\boldsymbol{P}^{\hat{\boldsymbol{s}}\hat{\boldsymbol{s}}})^{-1} \boldsymbol{s}_t + \hat{\boldsymbol{s}}^T (\boldsymbol{P}^{\hat{\boldsymbol{s}}\hat{\boldsymbol{s}}})^{-1} \hat{\boldsymbol{s}}. \end{aligned} \tag{32}$$

We can define the joint vector

$$\boldsymbol{z}_t \equiv \begin{bmatrix} \boldsymbol{s}_t \\ \boldsymbol{y}_t \end{bmatrix}$$

and assume it to be Gaussian

$$p\left(\boldsymbol{z}_t\right) \sim \mathcal{N}(\bar{\boldsymbol{z}}_t, \boldsymbol{P}^{\boldsymbol{z}\boldsymbol{z}})$$

with

$$\begin{aligned}
\bar{\boldsymbol{z}}_t &= \begin{bmatrix} \bar{\boldsymbol{x}}_t \\ \bar{\boldsymbol{y}}_t \end{bmatrix} = E\left(\begin{bmatrix} \boldsymbol{s}_t \\ \boldsymbol{y}_t \end{bmatrix} \mid \boldsymbol{s}_{t-1}, \boldsymbol{y}_{t-1}\right) = \begin{bmatrix} \hat{\boldsymbol{s}}_{t|t-1} \\ \hat{\boldsymbol{y}}_{t|t-1} \end{bmatrix} \\
\boldsymbol{P}^{\boldsymbol{zz}} &= E\left([\boldsymbol{z}_t - \bar{\boldsymbol{z}}][\boldsymbol{z}_t - \bar{\boldsymbol{z}}]^T \mid \boldsymbol{s}_{t-1}, \boldsymbol{y}_{1:t-1}\right) \\
&= \begin{bmatrix} \boldsymbol{P}^{\boldsymbol{ss}} & \boldsymbol{P}^{\boldsymbol{sy}} \\ \boldsymbol{P}^{\boldsymbol{ys}} & \boldsymbol{P}^{\boldsymbol{yy}} \end{bmatrix} \equiv \begin{bmatrix} \boldsymbol{P}^{\boldsymbol{ss}}_{t|t-1} & \boldsymbol{P}^{\boldsymbol{sy}}_{t|t-1} \\ \boldsymbol{P}^{\boldsymbol{ys}}_{t|t-1} & \boldsymbol{P}^{\boldsymbol{yy}}_{t|t-1} \end{bmatrix}.
\end{aligned}$$

The block inverse of $\boldsymbol{P}^{\boldsymbol{zz}}$ is

$$\begin{aligned}
(\boldsymbol{P}^{\boldsymbol{zz}})^{-1} &= \begin{bmatrix} \boldsymbol{B}_{11} & \boldsymbol{B}_{12} \\ \boldsymbol{B}_{12} & \boldsymbol{B}_{22} \end{bmatrix} \\
&= \begin{bmatrix} (\boldsymbol{P}^{\boldsymbol{ss}} - \boldsymbol{P}^{\boldsymbol{sy}}(\boldsymbol{P}^{\boldsymbol{yy}})^{-1}\boldsymbol{P}^{\boldsymbol{ys}})^{-1} & -\boldsymbol{B}_{11}\boldsymbol{P}^{\boldsymbol{sy}}(\boldsymbol{P}^{\boldsymbol{yy}})^{-1} \\ -\boldsymbol{B}_{22}\boldsymbol{P}^{\boldsymbol{ys}}(\boldsymbol{P}^{\boldsymbol{ss}})^{-1} & (\boldsymbol{P}^{\boldsymbol{yy}} - \boldsymbol{P}^{\boldsymbol{ys}}(\boldsymbol{P}^{\boldsymbol{ss}})^{-1}\boldsymbol{P}^{\boldsymbol{sy}})^{-1} \end{bmatrix}
\end{aligned}$$

and the joint density is given by

$$p\left(\boldsymbol{z}_t\right) = \frac{1}{(2\pi)^{(d_x+d_y)/2} \mid \boldsymbol{P}^{\boldsymbol{zz}} \mid^{1/2}} \exp\left(-\frac{1}{2}C\right)$$

with

$$\begin{aligned}
C &= \begin{bmatrix} (\boldsymbol{s}_t - \bar{\boldsymbol{x}}) & (\boldsymbol{y}_t - \bar{\boldsymbol{y}}) \end{bmatrix}^T (\boldsymbol{P}^{\boldsymbol{zz}})^{-1} \begin{bmatrix} (\boldsymbol{s}_t - \bar{\boldsymbol{x}}) & (\boldsymbol{y}_t - \bar{\boldsymbol{y}}) \end{bmatrix} \\
&= [\boldsymbol{s}_t - \bar{\boldsymbol{x}}]^T B_{11} [\boldsymbol{s}_t - \bar{\boldsymbol{x}}] + [\boldsymbol{s}_t - \bar{\boldsymbol{x}}]^T B_{12} [\boldsymbol{y}_t - \bar{\boldsymbol{y}}] + \\
&\quad [\boldsymbol{y}_t - \bar{\boldsymbol{y}}]^T B_{21} [\boldsymbol{s}_t - \bar{\boldsymbol{x}}] + [(\boldsymbol{y}_t - \bar{\boldsymbol{y}})]^T B_{22} [\boldsymbol{y}_t - \bar{\boldsymbol{y}}] \\
&= \boldsymbol{s}_t^T B_{11} \boldsymbol{s}_t + \boldsymbol{s}_t^T [-B_{11}\bar{\boldsymbol{x}} + B_{12}(\boldsymbol{y}_t - \bar{\boldsymbol{y}})] + ... \qquad (33)
\end{aligned}$$

Equating the first terms in equations (32) and (33) gives

$$\boldsymbol{P}^{\hat{\boldsymbol{s}}\hat{\boldsymbol{s}}} = B_{11}^{-1} = \boldsymbol{P}^{\boldsymbol{ss}} - \boldsymbol{P}^{\boldsymbol{sy}}(\boldsymbol{P}^{\boldsymbol{yy}})^{-1}\boldsymbol{P}^{\boldsymbol{ys}} \qquad (34)$$

and from the second terms we obtain

$$\begin{aligned}
(\boldsymbol{P}^{\hat{\boldsymbol{s}}\hat{\boldsymbol{s}}})^{-1}\hat{\boldsymbol{s}} &= B_{11}\bar{\boldsymbol{x}} - B_{12}(\boldsymbol{y}_t - \bar{\boldsymbol{y}}) \\
&= (\boldsymbol{P}^{\boldsymbol{ss}} - \boldsymbol{P}^{\boldsymbol{sy}}(\boldsymbol{P}^{\boldsymbol{yy}})^{-1}\boldsymbol{P}^{\boldsymbol{ys}})^{-1}\bar{\boldsymbol{x}} + \\
&\quad (\boldsymbol{P}^{\boldsymbol{ss}} - \boldsymbol{P}^{\boldsymbol{sy}}(\boldsymbol{P}^{\boldsymbol{yy}})^{-1}\boldsymbol{P}^{\boldsymbol{ys}})^{-1}\boldsymbol{P}^{\boldsymbol{sy}}(\boldsymbol{P}^{\boldsymbol{yy}})^{-1}(\boldsymbol{y}_t - \bar{\boldsymbol{y}})
\end{aligned}$$

which together with equation (34) is the solution to the filtering problem

$$\hat{\boldsymbol{s}} = \bar{\boldsymbol{s}} + \boldsymbol{P}^{\boldsymbol{sy}}(\boldsymbol{P}^{\boldsymbol{yy}})^{-1}(\boldsymbol{y}_t - \bar{\boldsymbol{y}}).$$

The Kalman gain

$$\boldsymbol{K}_t = \boldsymbol{P}^{\boldsymbol{sy}}(\boldsymbol{P}^{\boldsymbol{yy}})^{-1}$$

updates recursively the prediction density and provides two parameters characterizing the posterior density

$$\begin{aligned}\hat{\boldsymbol{s}}_{t|t} &= \hat{\boldsymbol{s}}_{t|t-1} + \boldsymbol{K}_t(\boldsymbol{y}_t - \bar{\boldsymbol{y}}) \\ \boldsymbol{P}^{\boldsymbol{ss}}_{t|t} &= \boldsymbol{P}^{\boldsymbol{ss}}_{t|t-1} - \boldsymbol{K}_t \boldsymbol{P}^{\boldsymbol{yy}}_{t|t-1} \boldsymbol{K}_t^T\end{aligned}$$

with

$$\begin{aligned}\boldsymbol{P}^{\boldsymbol{ss}}_{t|t-1} &= E\left([\boldsymbol{s}_{t|t-1} - \hat{\boldsymbol{s}}_{t|t-1}][\boldsymbol{s}_{t|t-1} - \hat{\boldsymbol{s}}_{t|t-1}]^T\right) \\ \boldsymbol{P}^{\boldsymbol{sy}}_{t|t-1} &= E\left([\boldsymbol{s}_{t|t-1} - \hat{\boldsymbol{s}}_{t|t-1}][\boldsymbol{y}_{t|t-1} - \hat{\boldsymbol{y}}_{t|t-1}]^T\right) \\ \boldsymbol{P}^{\boldsymbol{yy}}_{t|t-1} &= E\left([\boldsymbol{y}_{t|t-1} - \hat{\boldsymbol{y}}_{t|t-1}][\boldsymbol{y}_{t|t-1} - \hat{\boldsymbol{y}}_{t|t-1}]^T\right).\end{aligned}$$

4.3.2 Linear

The prediction density for a linear model with Gaussian shocks, a fixed state and measurement covariance matrix $\boldsymbol{Q}$ and $\boldsymbol{R}$ and the prior $p(\boldsymbol{s}_0) = \mathcal{N}(\boldsymbol{s}_0; \hat{\boldsymbol{s}}_0, \boldsymbol{P}_0^{ss})$ is characterized by its expected value

$$\hat{\boldsymbol{s}}_{t|t-1} = \bar{\boldsymbol{s}} + \boldsymbol{P}(\hat{\boldsymbol{s}}_{\boldsymbol{t-1|t-1}} - \bar{\boldsymbol{s}}).$$

and the covariance

$$\boldsymbol{P}^{ss}_{t|t-1} = \boldsymbol{P}\boldsymbol{P}^{ss}_{t-1|t-1}\boldsymbol{P}' + \boldsymbol{Q}$$

where $\boldsymbol{P}$ is the equilibrium state transition and $\bar{\boldsymbol{s}}$ the steady state. The observable innovation is

$$\boldsymbol{y}_t^i = \boldsymbol{y}_t - (\bar{\boldsymbol{M}} + \boldsymbol{M}\hat{\boldsymbol{s}}_{t|t-1})$$

where $\boldsymbol{M}$ and $\bar{\boldsymbol{M}}$ are the slope and constant matrix of the measurement equation in (13) and (14). The innovation covariance is

$$\boldsymbol{P}^{yy}_{t|t-1} = \boldsymbol{M}\boldsymbol{P}^{ss}_{t|t-1}\boldsymbol{M}' + \boldsymbol{R}$$

and the log likelihood contribution to the sample log likelihood is

$$l_t = -0.5(d_y \log(2\pi) + \log|\boldsymbol{P}^{yy}_{t|t-1}|) + \boldsymbol{y}_t^i(\boldsymbol{P}^{yy}_{t|t-1})^{-1}(\boldsymbol{y}_t^i)'.$$

The posterior can be calculated by

$$\begin{aligned}p(\boldsymbol{s}_{t|t}) &= \mathcal{N}(\boldsymbol{s}_{t|t}; \hat{\boldsymbol{s}}_{t|t}, \boldsymbol{P}^{ss}_{t|t}) \\ \hat{\boldsymbol{s}}_{t|t} &= \hat{\boldsymbol{s}}_{t|t-1} + \boldsymbol{P}^{ss}_{t|t-1}\boldsymbol{M}'(\boldsymbol{P}^{yy}_{t|t-1})^{-1} \\ \boldsymbol{P}^{ss}_{t|t} &= \boldsymbol{P}^{ss}_{t|t-1} - \boldsymbol{P}^{ss}_{t|t-1}\boldsymbol{M}'(\boldsymbol{P}^{yy}_{t|t-1})^{-1}\boldsymbol{M}\boldsymbol{P}^{ss}_{t|t-1}.\end{aligned}$$

4.3.3 Gaussian

This section derives the posterior density for the special case of additive and Gaussian shocks. It serves the purpose to work out where exactly integration is needed and the Smolyak operator can be of use. Since the prior and posterior are conditionally normal only the mean and covariance have to be updated. The state and measurement equations are given by

$$\begin{aligned} \boldsymbol{s}_t &= \boldsymbol{g}(\boldsymbol{s}_{t-1}) + \boldsymbol{e}_t \\ \boldsymbol{y}_t &= \boldsymbol{m}(\boldsymbol{s}_t) + \boldsymbol{\epsilon}_t \end{aligned} \tag{35}$$

with normal shock distributions $\boldsymbol{e}_t \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{Q}_t)$ and $\boldsymbol{\epsilon}_t \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{R}_t)$. The prior of the states is given by $\mathcal{N}(\boldsymbol{s}_0; \hat{\boldsymbol{s}}_0, \boldsymbol{P}_0^{\boldsymbol{ss}})$ and the predictive density of equation (35) is

$$p\left(\boldsymbol{s}_t \,|\, \boldsymbol{s}_{t-1}\right) = \mathcal{N}(\boldsymbol{s}_t; \boldsymbol{g}(\boldsymbol{s}_{t-1}), \boldsymbol{Q}_t).$$

The prediction step (28) of the general filter recursion can be written as

$$p\left(\boldsymbol{s}_t \,|\, \boldsymbol{y}_{1:t-1}\right) = \int \mathcal{N}(\boldsymbol{s}_t; \boldsymbol{g}(\boldsymbol{s}_{t-1}), \boldsymbol{Q}_t)\, p\left(\boldsymbol{s}_{t-1} \,|\, \boldsymbol{y}_{1:t-1}\right) d\boldsymbol{s}_{t-1}. \tag{36}$$

Since the expected value of a normally distributed variable is

$$E(\boldsymbol{t}) \equiv \hat{\boldsymbol{t}} = \int \boldsymbol{t}\mathcal{N}(\boldsymbol{t}; \boldsymbol{f}(\boldsymbol{s}), \boldsymbol{\Sigma})\, d\boldsymbol{t} = \boldsymbol{f}(\boldsymbol{s})$$

equation (36) can be used to derive the prior state mean

$$\begin{aligned} E(\boldsymbol{s}_t \,|\, \boldsymbol{y}_{1:t-1}) \equiv \hat{\boldsymbol{s}}_{t|t-1} &= \int \boldsymbol{s}_t p\left(\boldsymbol{s}_t \,|\, \boldsymbol{y}_{1:t-1}\right) d\boldsymbol{s}_t \\ &= \int \boldsymbol{s}_t \left(\int \mathcal{N}(\boldsymbol{s}_t; \boldsymbol{g}(\boldsymbol{s}_{t-1}), \boldsymbol{Q}_t)\, p\left(\boldsymbol{s}_{t-1} \,|\, \boldsymbol{y}_{1:t-1}\right) d\boldsymbol{s}_{t-1} \right) d\boldsymbol{s}_t \\ &= \int \left(\int \boldsymbol{s}_t \mathcal{N}(\boldsymbol{s}_t; \boldsymbol{g}(\boldsymbol{s}_{t-1}), \boldsymbol{Q}_t)\, d\boldsymbol{s}_t \right) p\left(\boldsymbol{s}_{t-1} \,|\, \boldsymbol{y}_{1:t-1}\right) d\boldsymbol{s}_{t-1} \\ &= \int \boldsymbol{g}(\boldsymbol{s}_{t-1})\, p\left(\boldsymbol{s}_{t-1} \,|\, \boldsymbol{y}_{1:t-1}\right) d\boldsymbol{s}_{t-1}. \end{aligned}$$

If the previous posterior density is

$$p\left(\boldsymbol{s}_{t-1} \,|\, \boldsymbol{y}_{1:t-1}\right) = \mathcal{N}(\boldsymbol{s}_{t-1}; \hat{\boldsymbol{s}}_{t-1|t-1}, \boldsymbol{P}_{t-1|t-1}^{\boldsymbol{ss}}) \tag{37}$$

the prediction density is

$$p\left(\boldsymbol{s}_t \,|\, \boldsymbol{y}_{t-1}\right) = \mathcal{N}(\boldsymbol{s}_t; \hat{\boldsymbol{s}}_{t|t-1}, \boldsymbol{P}_{t|t-1}^{\boldsymbol{ss}})$$

$$\hat{\boldsymbol{s}}_{t|t-1} = \int \boldsymbol{g}(\boldsymbol{s}_{t-1}) \qquad \mathcal{N}(\boldsymbol{s}_{t-1}; \hat{\boldsymbol{s}}_{t-1|t-1}, \boldsymbol{P}_{t-1|t-1}^{\boldsymbol{ss}})\, d\boldsymbol{s}_{t-1} \tag{38}$$

$$\begin{aligned} \boldsymbol{P}_{t|t-1}^{\boldsymbol{ss}} &= \int \boldsymbol{g}(\boldsymbol{s}_{t-1})\boldsymbol{g}^T(\boldsymbol{s}_{t-1}) \mathcal{N}(\boldsymbol{s}_{t-1}; \hat{\boldsymbol{s}}_{t-1|t-1}, \boldsymbol{P}_{t-1|t-1}^{\boldsymbol{ss}})\, d\boldsymbol{s}_{t-1} \\ &\quad + \boldsymbol{Q}_t - \hat{\boldsymbol{s}}_{t|t-1}\hat{\boldsymbol{s}}_{t|t-1}^T. \end{aligned} \tag{39}$$

The expected value of the observed variable is then given by

$$\begin{aligned}\hat{\boldsymbol{y}}_{t|t-1} &= \int \boldsymbol{y}_t \qquad \mathcal{N}(\boldsymbol{s}_t; \hat{\boldsymbol{s}}_{t|t-1}, \boldsymbol{P}^{\boldsymbol{ss}}_{t|t-1})\, d\boldsymbol{s}_t \\ &= \int \boldsymbol{m}(\boldsymbol{s}_t)\mathcal{N}(\boldsymbol{s}_t; \hat{\boldsymbol{s}}_{t|t-1}, \boldsymbol{P}^{\boldsymbol{ss}}_{t|t-1})\, d\boldsymbol{s}_t. \end{aligned} \tag{40}$$

With $\boldsymbol{\epsilon}^m_{t|t-1} \equiv \boldsymbol{m}(\boldsymbol{s}_t) - \hat{\boldsymbol{y}}_{t|t-1}$ the covariance of $\boldsymbol{y}_t$ is

$$\begin{aligned}\boldsymbol{P}^{\boldsymbol{yy}}_{t|t-1} &= E\left(\left[\boldsymbol{\epsilon}^m_{t|t-1}\right]\left[\boldsymbol{\epsilon}^m_{t|t-1}\right]^T\right) \\ &= \int \boldsymbol{m}(\boldsymbol{s}_t)\, \boldsymbol{m}^T(\boldsymbol{s}_t)\, \mathcal{N}(\boldsymbol{s}_t; \hat{\boldsymbol{s}}_{t|t-1}, \boldsymbol{P}^{\boldsymbol{ss}}_{t|t-1})\, d\boldsymbol{s}_t + \boldsymbol{H}_t - \hat{\boldsymbol{y}}_{t|t-1}\hat{\boldsymbol{y}}^T_{t|t-1}. \end{aligned} \tag{41}$$

The last section shows that the Kalman filter step is also useful for the nonlinear Gaussian model and we can calculate the covariance

$$\begin{aligned}\boldsymbol{P}^{\boldsymbol{sy}}_{t|t-1} &= E\left(\left[\boldsymbol{s}_t - \hat{\boldsymbol{s}}_{t|t-1}\right]\left[\boldsymbol{\epsilon}^m_{t|t-1}\right]^T\right) \\ &= \int \boldsymbol{s}_t \boldsymbol{m}^T(\boldsymbol{s}_t)\mathcal{N}(\boldsymbol{s}_t; \hat{\boldsymbol{s}}_{t|t-1}, \boldsymbol{P}^{\boldsymbol{ss}}_{t|t-1})\, d\boldsymbol{s}_t - \hat{\boldsymbol{s}}_{t|t-1}\hat{\boldsymbol{y}}^T_{t|t-1} \end{aligned} \tag{42}$$

in order to obtain the Kalman gain

$$\boldsymbol{K}_t = \boldsymbol{P}^{\boldsymbol{sy}}_{t|t-1}(\boldsymbol{P}^{\boldsymbol{yy}}_{t|t-1})^{-1}.$$

The recursion is closed by the filter step (29) and we obtain the next posterior density

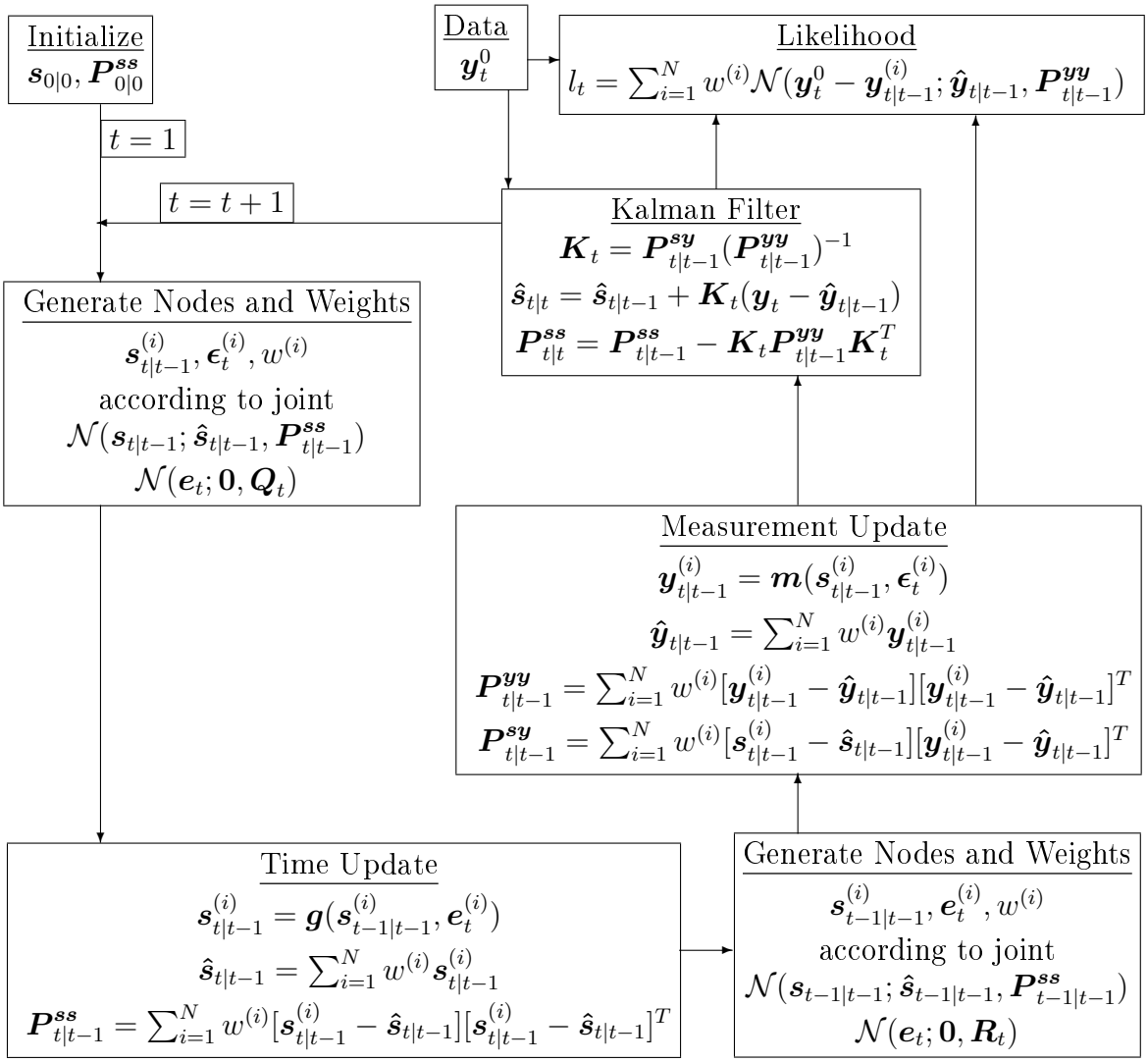
$$\begin{aligned} p\left(\boldsymbol{s}_t \,|\, \boldsymbol{y}_{1:t}\right) &= \mathcal{N}(\boldsymbol{s}_t; \hat{\boldsymbol{s}}_{t|t}, \boldsymbol{P}^{\boldsymbol{ss}}_{t|t}) \\ \hat{\boldsymbol{s}}_{t|t} &= \hat{\boldsymbol{s}}_{t|t-1} + \boldsymbol{K}_t[\boldsymbol{y}_t - \hat{\boldsymbol{y}}_{t|t-1}] \\ \boldsymbol{P}^{\boldsymbol{ss}}_{t|t} &= \boldsymbol{P}^{\boldsymbol{ss}}_{t|t-1} - \boldsymbol{K}_t\boldsymbol{P}^{\boldsymbol{yy}}_{t|t-1}\boldsymbol{K}^T_t. \end{aligned}$$

The mathematical problem and the only approximation needed for this filter is the evaluation of the integrals for the mean and covariance of the state prediction in equations (38) and (39), the expected value of the observables in equation (40), the innovation covariance in equation (41) and the covariance between the states and observations in equation (42). The integrals involved have the form

$$\begin{aligned}\mathcal{I}(\boldsymbol{f})_{\mathcal{N}} &= \int \boldsymbol{f}(\boldsymbol{s})\, \mathcal{N}(\boldsymbol{s}, \hat{\boldsymbol{s}}, \boldsymbol{P}^{\boldsymbol{ss}})\, d\boldsymbol{s} \\ &\approx \int \boldsymbol{f}(\boldsymbol{s}) \sum_{i=1}^{N} w^{(i)} \delta(\boldsymbol{s} - \boldsymbol{s}^{(i)})\, d\boldsymbol{s} = \sum_{i=1}^{N} w^{(i)}\, \boldsymbol{f}(\boldsymbol{s}^{(i)}) \end{aligned}$$

and can therefore be approximated by Gaussian quadrature by means of nodes $\boldsymbol{s}^{(j)}$ and weights $w^{(j)}$. Figure 8 shows the recursive steps for a general nonadditive noise model. With additive Gaussian noise only the integrals for two moments

Figure 8: Gaussian Filter



have to be approximated. For the general nonadditive model the approximation encompasses also the assumption of normally distributed prior and posterior densities. The integrands in (38) to (42) have then to be evaluated at the joint nodes and weights for states and shocks.
The approximation of the prediction mean and covariance in equations (38), (39)

for the additive shock model is given by

$$\begin{aligned}
\hat{\boldsymbol{s}}_{t|t-1} &= \sum_{i=1}^{N} w^{(i)} \boldsymbol{g}(\boldsymbol{s}_{t-1|t-1}^{(i)}) \\
\boldsymbol{P}_{t|t-1}^{\boldsymbol{ss}} &= \sum_{i=1}^{N} w^{(i)} \boldsymbol{g}(\boldsymbol{s}_{t-1|t-1}^{(i)}) \boldsymbol{g}^T(\boldsymbol{s}_{t-1|t-1}^{(i)}) + \boldsymbol{Q}_t - \hat{\boldsymbol{s}}_{t|t-1} \hat{\boldsymbol{s}}_{t|t-1}^T.
\end{aligned}$$

From the density $\mathcal{N}(\boldsymbol{s}_{t|t-1}; \hat{\boldsymbol{s}}_{t|t-1}, \boldsymbol{P}_{t|t-1}^{\boldsymbol{ss}})$ new nodes and weights are generated and the measurement update moments in equations (40), (41), (42) are calculated by

$$\begin{aligned}
\hat{\boldsymbol{y}}_{t|t-1} &= \sum_{i=1}^{N} w^{(i)} \boldsymbol{m}(\boldsymbol{s}_{t|t-1}^{(i)}) \\
\boldsymbol{P}_{t|t-1}^{\boldsymbol{yy}} &= \sum_{i=1}^{N} w^{(i)} \boldsymbol{m}(\boldsymbol{s}_{t|t-1}^{(i)}) \boldsymbol{m}^T(\boldsymbol{s}_{t|t-1}^{(i)}) + \boldsymbol{R} - \hat{\boldsymbol{y}}_{t|t-1} \hat{\boldsymbol{y}}_{t|t-1}^T \\
\boldsymbol{P}_{t|t-1}^{\boldsymbol{sy}} &= \sum_{i=1}^{N} w^{(i)} [\boldsymbol{s}_{t|t-1}^{(i)} - \hat{\boldsymbol{s}}_{t|t-1}][\boldsymbol{m}(\boldsymbol{s}_{t|t-1}^{(i)}) - \hat{\boldsymbol{y}}_{t|t-1}]^T.
\end{aligned}$$

4.4 Particle Filter

The last section assumed that the posterior density $p(\boldsymbol{s}_t \,|\, \boldsymbol{y}_{1:t})$ is Gaussian. It allows to generate nodes and weights and to update this discrete density approximation. For general nonlinear state space models and possibly non-Gaussian shocks the posterior is nonstandard or even multimodal. Without an analytical expression for these densities it is impossible to directly generate nodes and weights or random draws. Nevertheless there are indirect ways to generate random draws and the one usually used is importance sampling. Another random number generator class are Markov Chain Monte Carlo methods. They do not rely like importance sampling on a proposal density which should be similar to the target density. The price for this generality is a slow convergence towards a representative sample which makes them impractical for recursive filtering. Fortunately good proposal densities for importance sampling filtering can be generated.

The particle filter owes its name to the random draws generated from the posterior density we are interested in. Since we cannot draw particles from the posterior density $p(\boldsymbol{s}_t \,|\, \boldsymbol{y}_{1:t})$ directly we divide it by the proposal density $q(\boldsymbol{s}_t \,|\, \boldsymbol{y}_{1:t})$ and obtain the importance weights

$$w(\boldsymbol{s}_t) = \frac{p(\boldsymbol{s}_t \,|\, \boldsymbol{y}_{1:t})}{q(\boldsymbol{s}_t \,|\, \boldsymbol{y}_{1:t})}.$$

This allows to rewrite the general integral of equation (31) as

$$E(\boldsymbol{g}(\boldsymbol{s}_t)) \equiv \hat{\boldsymbol{g}}(\boldsymbol{s}_t) = \frac{\int \boldsymbol{g}(\boldsymbol{s}_t) w(\boldsymbol{s}_t) q(\boldsymbol{s}_t \,|\, \boldsymbol{y}_{1:t})\, d\boldsymbol{s}_t}{\int w(\boldsymbol{s}_t) q(\boldsymbol{s}_t \,|\, \boldsymbol{y}_{1:t})\, d\boldsymbol{s}_t}.$$

N particles drawn from the proposal density $\boldsymbol{s}_t^{(i)} \sim q(\boldsymbol{s}_t \,|\, \boldsymbol{y}_{1:t})$ and the associated weights represent an approximation to the posterior. The integral of interest can be calculated as

$$\hat{\boldsymbol{g}}(\boldsymbol{s}_t) = \frac{\frac{1}{N}\sum_{i=1}^{N} \boldsymbol{g}(\boldsymbol{s}_t^{(i)}) w(\boldsymbol{s}_t^{(i)})}{\frac{1}{N}\sum_{i=1}^{N} w(\boldsymbol{s}_t^{(i)})} = \sum_{i=1}^{N} \boldsymbol{g}(\boldsymbol{s}_t^{(i)}) \tilde{w}(\boldsymbol{s}_t^{(i)})$$

with normalized weights

$$\tilde{w}(\boldsymbol{s}_t^{(i)}) = \frac{w(\boldsymbol{s}_t^{(i)})}{\sum_{i=1}^{N} w(\boldsymbol{s}_t^{(i)})}.$$

According to the factorization in equations (29) and (28) for p and analogous for q the weight formula can be rewritten such that the weights are updated recursively

$$w(\boldsymbol{s}_t) = \frac{p\,(\boldsymbol{s}_t \,|\, \boldsymbol{y}_{1:t})}{q(\boldsymbol{s}_t \,|\, \boldsymbol{y}_{1:t})} = \frac{l_t^{-1} p\,(\boldsymbol{y}_t \,|\, \boldsymbol{s}_t)\, p\,(\boldsymbol{s}_t \,|\, \boldsymbol{y}_{1:t-1})}{q(\boldsymbol{s}_t \,|\, \boldsymbol{y}_{1:t})} \tag{43}$$

$$= \frac{l_t^{-1} p\,(\boldsymbol{y}_t \,|\, \boldsymbol{s}_t) \int p\,(\boldsymbol{s}_t \,|\, \boldsymbol{s}_{t-1})\, p\,(\boldsymbol{s}_{t-1} \,|\, \boldsymbol{y}_{1:t-1})\, d\boldsymbol{s}_{t-1}}{\int q(\boldsymbol{s}_t \,|\, \boldsymbol{s}_{t-1}) q(\boldsymbol{s}_{t-1} \,|\, \boldsymbol{y}_{1:t-1})\, d\boldsymbol{s}_{t-1}}. \tag{44}$$

4.4.1 Bootstrap

The discrete approximation of $p\,(\boldsymbol{s}_{t-1} \,|\, \boldsymbol{y}_{1:t-1})$ by a set of particles $\boldsymbol{s}_{t-1|t-1}^{(i)}$ and weights $w_{t-1}^{(i)}$ is given by

$$p\,(\boldsymbol{s}_{t-1} \,|\, \boldsymbol{y}_{1:t-1}) \approx \sum_{i=1}^{N} w_{t-1}^{(i)} \delta(\boldsymbol{s}_{t-1} - \boldsymbol{s}_{t-1|t-1}^{(i)}) \quad \text{with } w_{t-1}^{(i)} = \frac{p\,(\boldsymbol{s}_{t-1|t-1}^{(i)})}{q(\boldsymbol{s}_{t-1|t-1}^{(i)})}. \tag{45}$$

The next period particles $\boldsymbol{s}_{t|t-1}^{(i)}$ are obtained from equation (25)

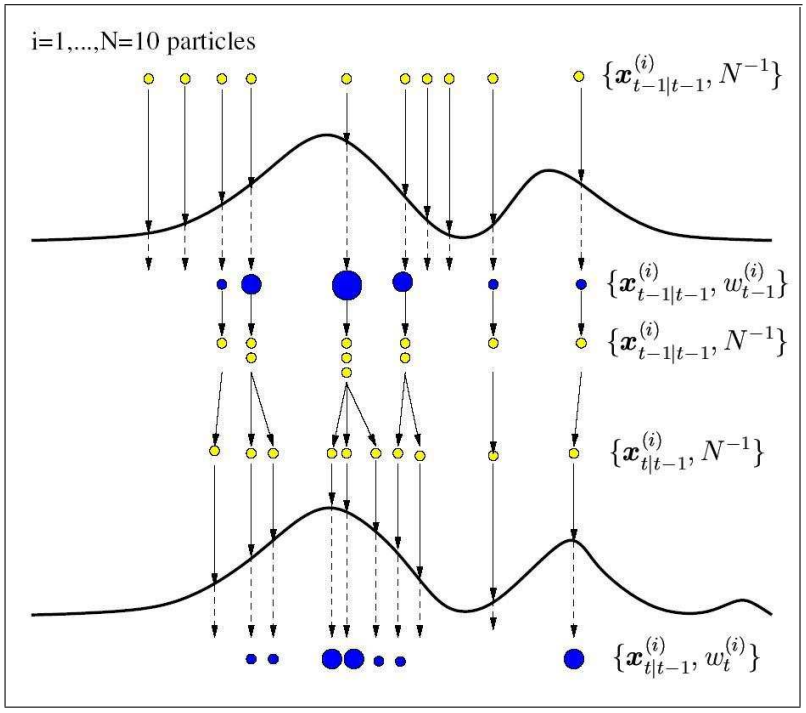
$$\boldsymbol{s}_{t|t-1}^{(i)} = \boldsymbol{g}(\boldsymbol{s}_{t-1|t-1}^{(i)}, \boldsymbol{e}_t^{(i)}) \tag{46}$$

where in addition draws from the state shock distribution $\boldsymbol{e}_t^{(i)} \sim q^e(\boldsymbol{e})$ have to be generated. Equation (45) allows to write the weight update equation (44) as

$$w_t^{(i)} \propto \frac{p\,(\boldsymbol{y}_t \,|\, \boldsymbol{s}_{t|t-1}^{(i)})\, p\,(\boldsymbol{s}_{t|t-1}^{(i)} \,|\, \boldsymbol{s}_{t-1|t-1}^{(i)})\, p\,(\boldsymbol{s}_{t-1|t-1}^{(i)})}{q(\boldsymbol{s}_{t|t-1}^{(i)} \,|\, \boldsymbol{s}_{t-1|t-1}^{(i)}) q(\boldsymbol{s}_{t-1|t-1}^{(i)})}$$

$$= w_{t-1}^{(i)} \frac{p\,(\boldsymbol{y}_t \,|\, \boldsymbol{s}_{t|t-1}^{(i)})\, p\,(\boldsymbol{s}_{t|t-1}^{(i)} \,|\, \boldsymbol{s}_{t-1|t-1}^{(i)})}{q(\boldsymbol{s}_{t|t-1}^{(i)} \,|\, \boldsymbol{s}_{t-1|t-1}^{(i)})}.$$

Figure 9: Resampling Scheme



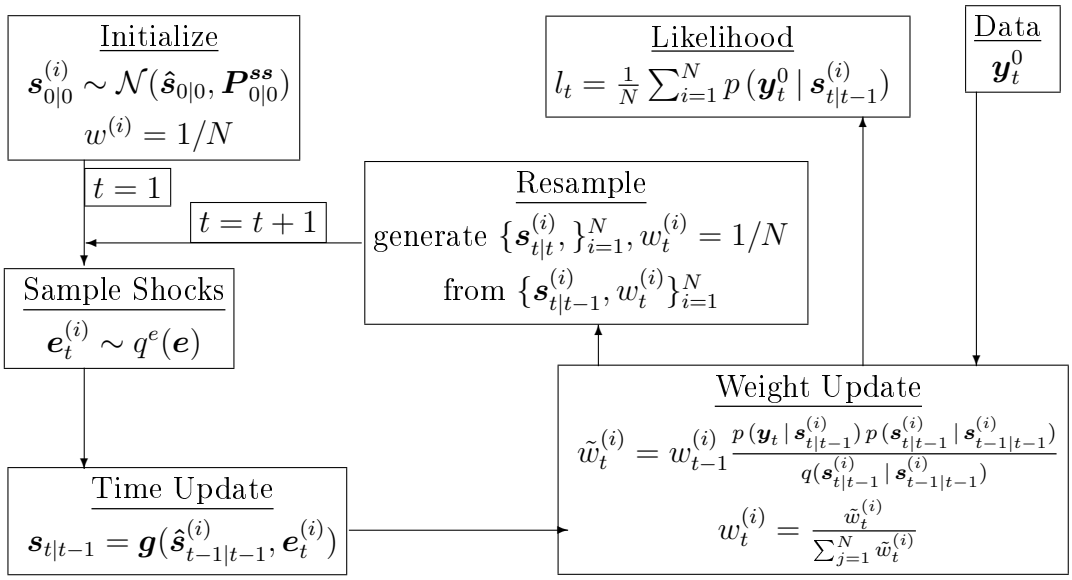
Source: van der Merwe, de Freitas, Doucet, and Wan (2001)

The posterior is finally approximated by the discrete density

$$p\left(\boldsymbol{s}_t \mid \boldsymbol{y}_{1:t}\right) \approx \sum_{i=1}^{N} w_t^{(i)} \delta(\boldsymbol{s}_t - \boldsymbol{s}_{t|t}^{(i)}).$$

Where do we get the posterior particles $\boldsymbol{s}_{t|t}^{(i)}$ from? A serious problem with this filter is that a recursive update of the state particles in equation (46) increases their variance without bound as $t \to \infty$. The particles move away from the expected value and their weight degenerates to zero. A brute force solution is to use more particles. A more elegant solution was proposed by Gordon, Salmond, and Smith (1993) and resamples the particles at each step according to their weight. The particles with low weights are dropped and the ones with high weights are duplicated. These resampled particles represent an equally weighted posterior sample $\boldsymbol{s}_{t|t}^{(i)}$. This procedure is shown in figure 9 and the general particle filter is summarized in figure 10.
The implementation of an importance sampling particle filter needs the specification of a proposal density $q(\boldsymbol{s}_{t|t-1} \mid \boldsymbol{s}_{t-1|t-1})$ for the evaluation of probabilities of the importance weights in equation (47). Moreover the probabilities $p\left(\boldsymbol{y}_t \mid \boldsymbol{s}_{t|t-1}\right)$ and $p\left(\boldsymbol{s}_{t|t-1} \mid \boldsymbol{s}_{t-1|t-1}\right)$ are needed. They can be difficult to compute if shocks

Figure 10: Bootstrap Filter



are nonadditive and non-Gaussian and inverse densities have to be evaluated to obtain a likelihood value.
The simplest Monte Carlo variant of a particle filter, called bootstrap or sequential importance resampling particle filter, takes the state transition as the proposal density

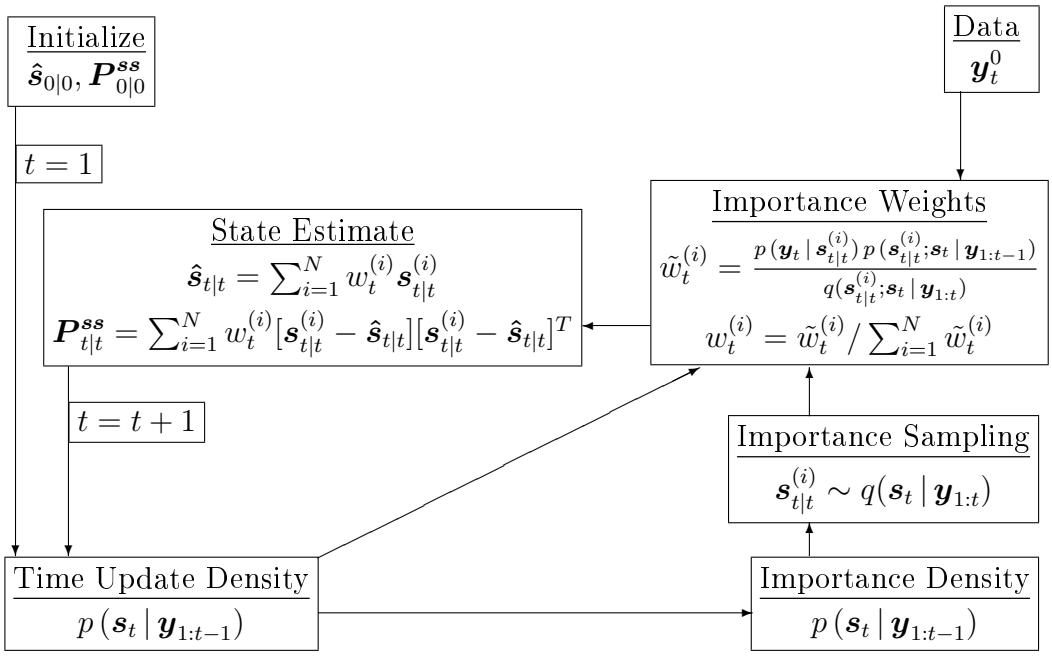
$$q(s_{t|t-1}^{(i)} \mid s_{t-1|t-1}^{(i)}) = p(s_{t|t-1}^{(i)} \mid s_{t-1|t-1}^{(i)}).$$

This simplifies the weight update equation (44) and we obtain

$$w_t^{(i)} = w_{t-1}^{(i)} p(y_t \mid s_{t|t-1}^{(i)}).$$

The disadvantage is that the last observation y_t is not taken into account in the importance density to form the posterior. The consequence is that we may sample in low probability regions of the true posterior with low probabilities of the particles. This decreases the effective number of particles and more particles or the resampling step are needed.
The principle of the bootstrap filter is a trial and error approach. Starting from the prior density of the unobservables, one simulates many shock realizations with the given shock variance. This gives the next period state realizations and together with simulated measurement shocks we get many simulated observables. Many of the simulated observables will be very unlikely. These observables and their generating unobservables are dropped for simulations in the following periods. The remaining particles of the unobservables can be taken as an estimate of their density. The bootstrap filter can be interpreted as a genetic algorithm where the resampling step governs the survival of the fittest.

Figure 11: General Particle Filter



Fernádez-Villaverde and Rubio-Ramírez (2004b) use this bootstrap filter with 40,000 particles. For each particle and observation the policy for the next period has to be interpolated in the prediction step. This step is a main bottleneck in the whole likelihood approach. In case of a linear finite element approximation this step is a fast table look up to locate the involved subdomain combined with a linear interpolation. In case of spectral approximation the interpolation is more expensive. The price for the reduction of the number of nodes needed for a spectral approximation is therefore an increased effort in the interpolation step within the likelihood evaluation.

4.4.2 Gaussian

Particle filters without resampling rely on equation (43). For a sample $\boldsymbol{s}^{(i)}_{t|t} \sim q(\boldsymbol{s}_{t|t}, \boldsymbol{y}_{1:t})$ from the importance density the importance weights are

$$w^{(i)}_t \propto \tilde{w}^{(i)}_t = \frac{p(\boldsymbol{y}_t \mid \boldsymbol{s}^{(i)}_{t|t})\, p(\boldsymbol{s}^{(i)}_{t|t}; \boldsymbol{s}_t \mid \boldsymbol{y}_{1:t-1})}{q(\boldsymbol{s}^{(i)}_{t|t}; \boldsymbol{s}_t \mid \boldsymbol{y}_{1:t})} \qquad w^{(i)}_t = \frac{\tilde{w}^{(i)}_t}{\sum_{i=1}^N \tilde{w}^{(i)}_t}$$

where $p(x^{(i)}; x \mid y)$ is the density of x evaluated at $x^{(i)}$. This general particle filter is shown in figure 11. The Gaussian particle filter by Kotecha and Djurić (2003a) starts with an approximation to the previous period posterior by a Gaussian

density

$$p\left(\boldsymbol{s}_{t-1} \,|\, \boldsymbol{y}_{1:t-1}\right) \approx \mathcal{N}(\boldsymbol{s}_{t-1}; \hat{\boldsymbol{s}}_{t-1|t-1}, \boldsymbol{P}^{\boldsymbol{ss}}_{t-1|t-1})$$

and generates nodes $\boldsymbol{s}^{(i)}_{t-1|t-1}$ and weights w_i according to this law. The nodes are updated to $\boldsymbol{s}^{(i)}_{t|t-1}$ by the state equation (46) and the prior density is then approximated by

$$\begin{aligned} p\left(\boldsymbol{s}_t \,|\, \boldsymbol{y}_{1:t-1}\right) &\approx \mathcal{N}(\boldsymbol{s}_t; \hat{\boldsymbol{s}}_{t|t-1}, \boldsymbol{P}^{\boldsymbol{ss}}_{t|t-1}) \\ \hat{\boldsymbol{s}}_{t|t-1} &= \sum_{i=1}^{N} w_i \boldsymbol{s}^{(i)}_{t|t-1} \\ \boldsymbol{P}^{\boldsymbol{ss}}_{t|t-1} &= \sum_{i=1}^{N} w_i [\boldsymbol{s}^{(i)}_{t|t-1} - \hat{\boldsymbol{s}}_{t|t-1}][\boldsymbol{s}^{(i)}_{t|t-1} - \hat{\boldsymbol{s}}_{t|t-1}]^T. \end{aligned}$$

Particles $\boldsymbol{s}^{(i)}_{t|t}$ are drawn from the importance density and by

$$w^{(i)}_t \propto \frac{p\left(\boldsymbol{y}_t \,|\, \boldsymbol{s}^{(i)}_{t|t}\right) \mathcal{N}(\boldsymbol{s}^{(i)}_{t|t}; \hat{\boldsymbol{s}}_{t|t-1}, \boldsymbol{P}^{\boldsymbol{ss}}_{t|t-1})}{q(\boldsymbol{s}^{(i)}_{t|t}; \boldsymbol{s}_t \,|\, \boldsymbol{y}_{1:t-1})}$$

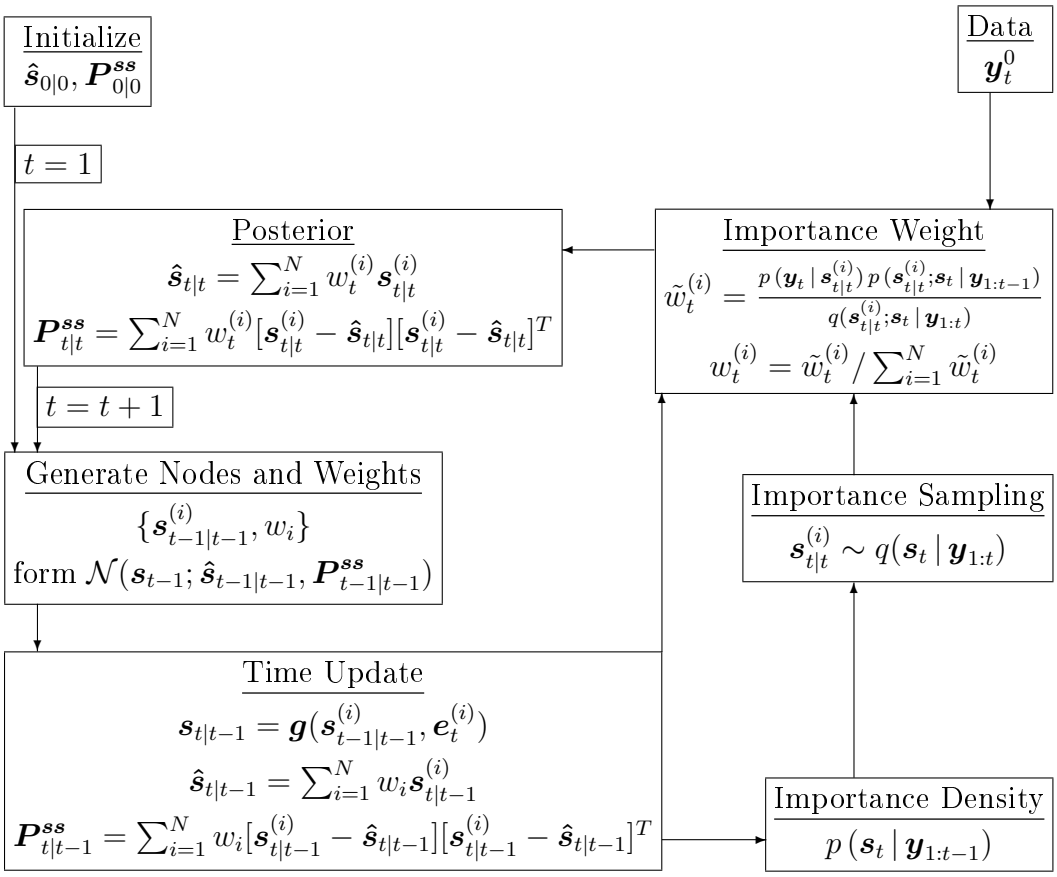
their weights are calculated. Finally the approximation of the posterior is calculated by

$$\begin{aligned} p\left(\boldsymbol{s}_{t|t} \,|\, \boldsymbol{y}_{1:t}\right) &\approx \mathcal{N}(\boldsymbol{s}_t; \hat{\boldsymbol{s}}_{t|t}, \boldsymbol{P}^{\boldsymbol{ss}}_{t|t}) \\ \hat{\boldsymbol{s}}_{t|t} &= \sum_{i=1}^{N} w^{(i)}_t \boldsymbol{s}^{(i)}_{t|t} \\ \boldsymbol{P}^{\boldsymbol{ss}}_{t|t} &= \sum_{i=1}^{N} w^{(i)}_t [\boldsymbol{s}^{(i)}_{t|t} - \hat{\boldsymbol{s}}_{t|t}][\boldsymbol{s}^{(i)}_{t|t} - \hat{\boldsymbol{s}}_{t|t}]^T. \end{aligned}$$

This filter is the basis of a more general filter provided in Kotecha and Djurić (2003b) as a companion paper. It is a generalization towards a more accurate approximation of the posterior compared to a simple Gaussian density. The approximation is constructed as a sum of Gaussian densities to approximate more than just the first two moments of the posteriors. The moment update in the basic Gaussian particle filter is shown in figure 12. The more general filter is yet not implemented but it could be an an interesting next step after the Gaussian filters are tested.

The Gaussian particle filter still lacks the specification of an importance density. One approach is to combine it with one of the nonlinear Gaussian filters. van der Merwe, de Freitas, Doucet, and Wan (2001) combined the unscented Kalman filter with the bootstrap filter and improved the performance substantially. I

Figure 12: Gaussian Particle



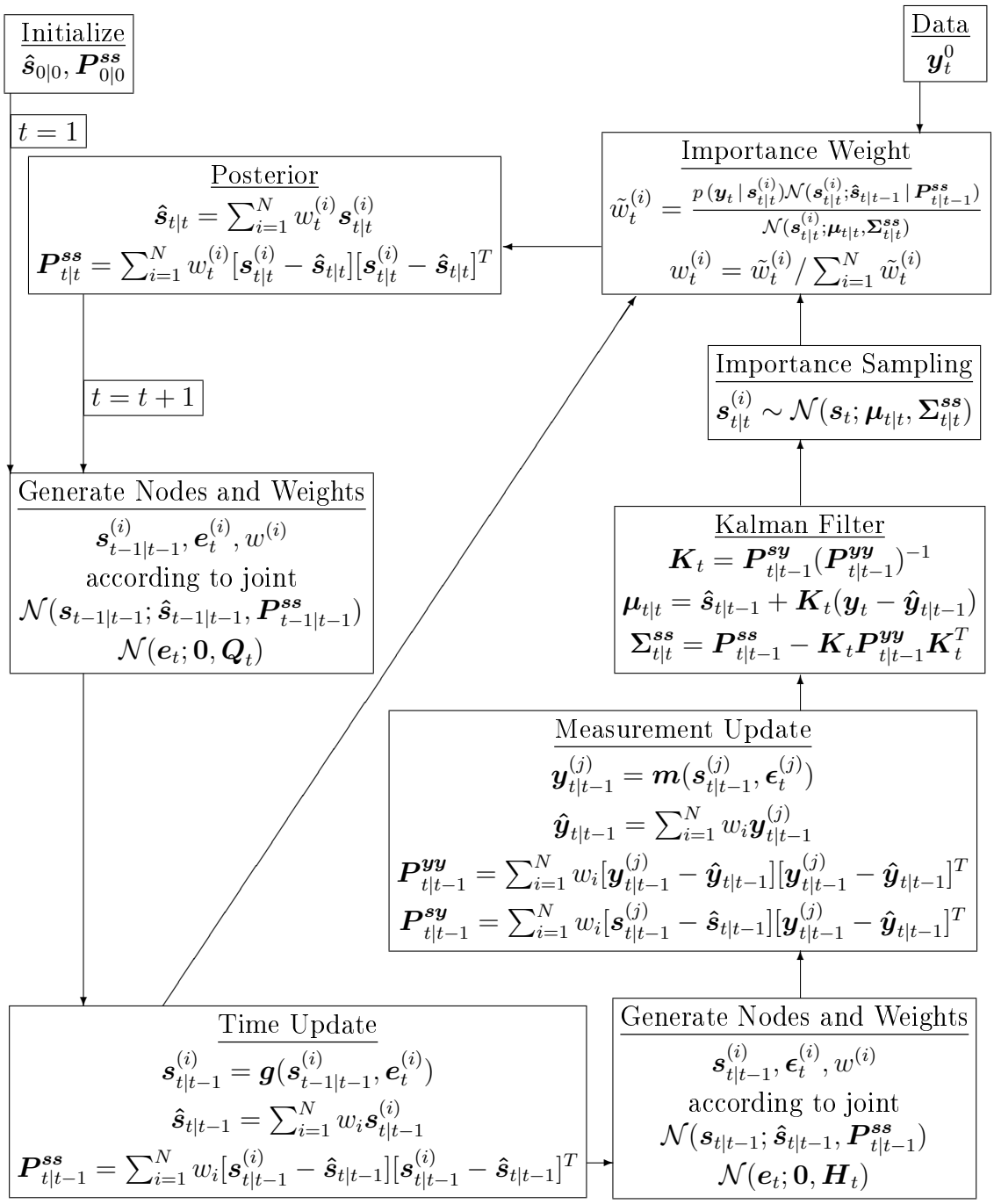
implemented the Gaussian particle filter where the Gaussian filter provides the importance density.
As in the Gaussian filter the prior is approximated by a normal density

$$p(\boldsymbol{s}_t \mid \boldsymbol{y}_{1:t-1}) \approx \mathcal{N}(\boldsymbol{s}_t; \hat{\boldsymbol{s}}_{t|t-1}, \boldsymbol{P}^{\boldsymbol{ss}}_{t|t-1})$$

and the prior particles and their moments $\hat{\boldsymbol{s}}_{t|t-1}, \boldsymbol{P}^{\boldsymbol{ss}}_{t|t-1}$ are calculated by the time update

$$\begin{aligned}
\boldsymbol{s}_{t|t-1} &= \boldsymbol{g}(\boldsymbol{s}_{t-1|t-1}^{(i)}, \boldsymbol{e}_t^{(i)}) \\
\hat{\boldsymbol{s}}_{t|t-1} &= \sum_{i=1}^N w_{t-1}^{(i)} \boldsymbol{s}_{t|t-1}^{(i)} \\
\boldsymbol{P}^{\boldsymbol{ss}}_{t|t-1} &= \sum_{i=1}^N w_{t-1}^{(i)} [\boldsymbol{s}_{t|t-1}^{(i)} - \hat{\boldsymbol{s}}_{t|t-1}][\boldsymbol{s}_{t|t-1}^{(i)} - \hat{\boldsymbol{s}}_{t|t-1}]^T.
\end{aligned}$$

Figure 13: Gaussian Particle Filter



For this density nodes $\boldsymbol{s}^{(j)}_{t|t-1}$ and weights $w^{(j)}$ are generated

$\boldsymbol{s}^{(j)}_{t|t-1} \sim \mathcal{N}(\boldsymbol{s}_t; \hat{\boldsymbol{s}}_{t|t-1}, \boldsymbol{P}^{\boldsymbol{ss}}_{t|t-1})$

and used for the measurement update

$$\begin{aligned}
\boldsymbol{y}_{t|t-1}^{(j)} &= \boldsymbol{m}(\boldsymbol{s}_{t|t-1}^{(j)}, \boldsymbol{\epsilon}_t^{(j)}) \\
\hat{\boldsymbol{y}}_{t|t-1} &= \sum_{i=1}^{N} w_j \boldsymbol{y}_{t|t-1}^{(j)}
\end{aligned}$$

$$\begin{aligned}
\boldsymbol{P}_{t|t-1}^{\boldsymbol{yy}} &= \sum_{i=1}^{N} w_j [\boldsymbol{y}_{t|t-1}^{(j)} - \hat{\boldsymbol{y}}_{t|t-1}][\boldsymbol{y}_{t|t-1}^{(j)} - \hat{\boldsymbol{y}}_{t|t-1}]^T \\
\boldsymbol{P}_{t|t-1}^{\boldsymbol{sy}} &= \sum_{i=1}^{N} w_j [\boldsymbol{s}_{t|t-1}^{(j)} - \hat{\boldsymbol{s}}_{t|t-1}][\boldsymbol{y}_{t|t-1}^{(j)} - \hat{\boldsymbol{y}}_{t|t-1}]^T
\end{aligned}$$

to obtain the importance density via the Kalman gain

$$\begin{aligned}
\boldsymbol{K}_t &= \boldsymbol{P}_{t|t-1}^{\boldsymbol{sy}} (\boldsymbol{P}_{t|t-1}^{\boldsymbol{yy}})^{-1} \\
q(\boldsymbol{s}_t \,|\, \boldsymbol{y}_{1:t}) &= \mathcal{N}(\boldsymbol{s}_t; \boldsymbol{\mu}_{t|t}, \boldsymbol{\Sigma}_{t|t}^{\boldsymbol{ss}}) \\
\boldsymbol{\mu}_{t|t} &= \hat{\boldsymbol{s}}_{t|t-1} + \boldsymbol{K}_t (\boldsymbol{y}_t - \hat{\boldsymbol{y}}_{t|t-1}) \\
\boldsymbol{\Sigma}_{t|t}^{\boldsymbol{ss}} &= \boldsymbol{P}_{t|t-1}^{\boldsymbol{ss}} - \boldsymbol{K}_t \boldsymbol{P}_{t|t-1}^{\boldsymbol{yy}} \boldsymbol{K}_t^T.
\end{aligned}$$

The importance density is used to generate particles

$$\boldsymbol{s}_{t|t}^{(i)} \sim \mathcal{N}(\boldsymbol{s}_t; \boldsymbol{\mu}_{t|t}, \boldsymbol{\Sigma}_{t|t}^{\boldsymbol{ss}})$$

and to calculate the associate importance weights

$$\begin{aligned}
\tilde{w}_t^{(i)} &= \frac{p\left(\boldsymbol{y}_t \,|\, \boldsymbol{s}_{t|t}^{(i)}\right) \mathcal{N}(\boldsymbol{s}_t; \hat{\boldsymbol{s}}_{t|t-1}, \boldsymbol{P}_{t|t-1}^{\boldsymbol{ss}})}{\mathcal{N}(\boldsymbol{s}_t; \boldsymbol{\mu}_{t|t}, \boldsymbol{\Sigma}_{t|t}^{\boldsymbol{ss}})} \\
w_t^{(i)} &= \tilde{w}_t^{(i)} \sum_{i=1}^{N} \tilde{w}_t^{(i)}
\end{aligned}$$

which determine the posterior through its moments

$$\begin{aligned}
p\left(\boldsymbol{s}_t \,|\, \boldsymbol{y}_{1:t}\right) &\approx \mathcal{N}(\boldsymbol{s}_t; \hat{\boldsymbol{s}}_{t|t}, \boldsymbol{P}_{t|t}^{\boldsymbol{ss}}) \\
\hat{\boldsymbol{s}}_{t|t} &= \sum_{i=1}^{N} w_t^{(i)} \boldsymbol{s}_{t|t}^{(i)} \\
\boldsymbol{P}_{t|t}^{\boldsymbol{ss}} &= \sum_{i=1}^{N} w_t^{(i)} [\boldsymbol{s}_{t|t}^{(i)} - \hat{\boldsymbol{s}}_{t|t}][\boldsymbol{s}_{t|t}^{(i)} - \hat{\boldsymbol{s}}_{t|t}]^T.
\end{aligned}$$

This filter is summarized in figure 13.

5 Posterior Density

In the Bayesian framework information accumulation is described by the Bayes formula. The basis for inference and object of interest is the posterior of the unobservables

$$p(\boldsymbol{\theta} \mid \boldsymbol{y}) = \frac{p(\boldsymbol{y} \mid \boldsymbol{\theta})\, p(\boldsymbol{\theta})}{p(\boldsymbol{y})} = \frac{p(\boldsymbol{y} \mid \boldsymbol{\theta})\, p(\boldsymbol{\theta})}{\int p(\boldsymbol{y} \mid \boldsymbol{\theta})\, p(\boldsymbol{\theta})\, d\boldsymbol{\theta}} \propto p(\boldsymbol{y} \mid \boldsymbol{\theta})\, p(\boldsymbol{\theta}).$$

It represents information about the unobservables $\boldsymbol{\theta}$ after available data is processed. $p(\boldsymbol{y})$ is the marginal likelihood and at the heart of the Bayesian model selection in section 6. The researcher's or the application specific utility function determines a point estimate from the posterior density. This is a feature which is not automatically embedded in the classical approach although the optimal point estimator depends on the utility function and risk aversion in a given application. An analytical expression is neither available for the likelihood nor for the posterior. The object of interest is a density and an approximation can be obtained by a random number generator.

5.1 Metropolis-Hastings

The Metropolis-Hastings algorithm is a subspecies of a Markov Chain Monte Carlo algorithm and allows to generate draws from any target density $p(\boldsymbol{x})$. As opposed to importance sampling no proposal density is needed. The only prerequisite is that the target density can be evaluated at any point $\boldsymbol{x}$ of its domain. The term Markov Chain refers to the fact that draws from the target density are Markovian and not independent. The algorithm is constructed as described in Chib and Greenberg (1995) in such a way that the density of the sequence of draws is the density of interest.
This algorithm is used to draw a sequence $\{\hat{\boldsymbol{\theta}}_n\}_{n=1}^{N}$ of structural parameter vectors from its posterior density so that for large N the sequence is distributed according to the posterior. It is then approximated by a histogram. The algorithm is summarized in table 3.
We start with a vector of structural parameters $\hat{\boldsymbol{\theta}}_0$ and draw a candidate vector $\hat{\boldsymbol{\theta}}_n^*$. In the basic Metropolis-Hastings algorithm a candidate is generated by a random walk with $\mathcal{N}(\hat{\boldsymbol{\theta}}_n^*; \hat{\boldsymbol{\theta}}_{n-1}, \boldsymbol{\Sigma}_{\boldsymbol{\epsilon}})$. Vector $\hat{\boldsymbol{\theta}}_0$ is the first member of the sequence of draws from the posterior. For $\hat{\boldsymbol{\theta}}_0$ and the candidate vectors the posterior kernel is calculated by evaluating the prior and likelihood. In step 2 (b) the ratio of these two posterior values is calculated. The candidate parameter vector is accepted as the second member of the sequence if the ratio of candidate to the last vector posterior is higher than a uniformly distributed random number between zero and one. The chance of the candidate vector to be accepted increases with the posterior ratio. If the candidate posterior value is higher than the last value the parameter vector will be accepted for sure since the uniform random number

Table 3: Metropolis-Hastings Algorithm

1. choose $\hat{\boldsymbol{\theta}}_0$, N and $\boldsymbol{\Sigma}_{\boldsymbol{\epsilon}}$ such that acceptance ratio is $\approx 30\%$
2. repeat the following steps starting with $n = 1$
   (a) generate $\hat{\boldsymbol{\theta}}_n^* = \hat{\boldsymbol{\theta}}_{n-1} + \boldsymbol{\epsilon}$, where $\mathcal{N}(\boldsymbol{\epsilon}; \mathbf{0}, \boldsymbol{\Sigma}_{\boldsymbol{\epsilon}})$
   (b) $\hat{\boldsymbol{\theta}}_n = \begin{cases} \hat{\boldsymbol{\theta}}_n^* & \text{if } U(0,1) \leq \frac{p\,(\boldsymbol{y}_{1:t}^0 \,|\, \hat{\boldsymbol{\theta}}_n^*)\, p\,(\hat{\boldsymbol{\theta}}_n^*)}{p\,(\boldsymbol{y}_{1:t}^0 \,|\, \hat{\boldsymbol{\theta}}_{n-1})\, p\,(\hat{\boldsymbol{\theta}}_{n-1})} \\ \hat{\hat{\boldsymbol{\theta}}}_{n-1} & \text{otherwise} \end{cases}$
   (c) calculate diagnostic test, choose J
   (d) if $n < N$, $n = n + 1$, goto 2 (a)
3. disregard burn-in draws $\hat{\boldsymbol{\theta}}_1, ..., \hat{\boldsymbol{\theta}}_J$

is one at maximum. But even for lower posterior ratios there is a chance for the candidate to be accepted since the random number can be even lower than the ratio. If the candidate vector is not accepted then the new draw from the posterior density is taken to be the last parameter vector. Again a candidate vector is constructed by a random walk step. This procedure and an acceptance ratio around 0.3 ensure that the sequence of accepted and repeated parameter vectors is distributed according to the true posterior. The acceptance ratio is the ratio of accepted candidates to all generated candidates.
The Metropolis-Hastings algorithm is also a global maximization procedure since it walks through the feasible parameter space and each step is guided by the relative fit in terms of the posterior value. If the prior is flat for all parameters the posterior is proportional to the likelihood and the Metropolis-Hastings algorithm will find the maximum of the likelihood.
The critical choices of the algorithm are the density to generate candidates $\hat{\boldsymbol{\theta}}_n^*$, the starting value $\hat{\boldsymbol{\theta}}_0$ and the number of draws N. The choice of $\hat{\boldsymbol{\theta}}_0$ determines the number of draws needed before convergence of the sequence is detected. The start value might be very far from a representative draw of the target density and many draws are needed to get into a representative region. But when is it representative or equivalently how long should be the burn-in sequence? This will be discussed in the next section about convergence diagnostic tests.
The distributional choice is often a random walk with normal shocks. For a normal target density the optimal choice of the innovation variance is is $\boldsymbol{\Sigma}_\epsilon = \text{Cov}(\boldsymbol{x})$. It has to be scaled so that the acceptance ratio is around 0.3. For a normal target density this is achieved by $\gamma^{RW} = 2.38/\sqrt{D}$. Of course the target density and its covariance $\text{Cov}(\boldsymbol{x})$ are not known as it is the object of

interest of the algorithm. In practice only the diagonal of the matrix $\boldsymbol{\Sigma}_\epsilon$ is tuned. This variance choice influences the region covered by the sequences. Sampling around the mode of the posterior with large variances will generate candidates far from the current value and therefore a low acceptance probability. Smaller variances increases the acceptance ratio with a too small region being covered so that low probability regions are undersampled. The recommended acceptance ratio results from the attempt to balance this trade off. Both a too high and too low variances will end up in high and slowly decreasing autocorrelations of the individual parameter sequences. A convergence test is therefore an important part of the analysis.

5.2 Convergence Test

To detect convergence one can either check several parallel sequences or divide one sequence into subsequences. Examining only one subdivided sequence will result in overly optimistic diagnostic tests. Gelman and Rubin (1992) pointed out that lack of convergence, in many problems, can easily be detected from many but not from one sequence.
Either one sequence is divided into two sequences or several sequences are generated from different start values, in both cases the diagnostic test is calculated for a three dimensional tensor $\hat{\boldsymbol{\theta}}$ of size $N \times D \times M$ with elements $\hat{\theta}^d_{n,m}$. D is the number of parameters, N the number of draws and M the number of sequences. $\hat{\boldsymbol{\theta}}_{n,m}$ is a $1 \times D$ vector and represents the n^{th} draw in the m^{th} sequence and $\hat{\boldsymbol{\theta}}_{:,m}$ is a $N \times D$ matrix and represents all draws in sequence m.
Brooks and Gelman (1998) proposed the multivariate potential scale reduction factor R as a diagnostic test. The general idea is to inspect within and between sequence variances and diagnose convergence if they are close to each other. The within sequence variance is the $D \times D$ matrix

$$\boldsymbol{W} = \frac{1}{M(N-1)} \sum_{m=1}^{M} \sum_{n=1}^{N} (\hat{\boldsymbol{\theta}}_{n,m} - \bar{\boldsymbol{\theta}}_m)'(\hat{\boldsymbol{\theta}}_{n,m} - \bar{\boldsymbol{\theta}}_m)$$

where $\bar{\boldsymbol{\theta}}_m = \frac{1}{N} \sum_{n=1}^{N} \hat{\boldsymbol{\theta}}_{n,m}$ is the $1 \times D$ mean vector in sequence m. $\boldsymbol{W}$ is the mean of the variances in each sequence. The between sequence variance $\boldsymbol{B}/N$ is the $D \times D$ matrix

$$\boldsymbol{B}/N = \frac{1}{M-1} \sum_{m=1}^{M} (\bar{\boldsymbol{\theta}}_m - \bar{\boldsymbol{\theta}})'(\bar{\boldsymbol{\theta}}_m - \bar{\boldsymbol{\theta}})$$

where $\bar{\boldsymbol{\theta}} = \frac{1}{M} \sum_{m=1}^{M} \bar{\boldsymbol{\theta}}_m$ is the $1 \times D$ mean of all draws. The combined variance can be estimated as

$$\boldsymbol{V} = \frac{N-1}{N} \boldsymbol{W} + (1 + \frac{1}{M}) \boldsymbol{B}/N.$$

Convergence is detected for similar $\boldsymbol{V}$ and $\boldsymbol{W}$. A distance measure represents the multivariate potential scale reduction factor

$$R = \frac{N-1}{N} + \frac{M+1}{M}\lambda_{max} \qquad \text{where } \lambda_{max} = \max_{\boldsymbol{a}} \frac{\boldsymbol{a}'\boldsymbol{V}\boldsymbol{a}}{\boldsymbol{a}'\boldsymbol{W}\boldsymbol{a}}.$$

λ_{max} can be calculated by taking the largest absolute eigenvalue of $\boldsymbol{W}^{-1}\boldsymbol{B}/N$. There are three conditions for convergence

- $\boldsymbol{V}$ and $\boldsymbol{W}$ should stabilize as a function of n,
- R should be below 1.1.

I calculate these numbers repeatedly after some draws and once the conditions are met the burn-in sequence length J is found. The draws thereafter are taken to represent draws from the posterior of structural parameters. My experience so far is that the first two criteria are met before the third. In the estimations I therefore generate burn-in draws until R is below 1.1.

5.3 Genetic Extension

The variances on the diagonal of matrix $\gamma^{RW}\boldsymbol{\Sigma}_\epsilon$ for the random walk innovations have to be chosen such that an acceptance ratio of around 0.3 is obtained. In the model at hand there are 10 parameters to be estimated and therefore 10 diagonal have to be tuned. To find good values simultaneously is quite demanding and many draws in several trial sequences have to be generated. The necessary number of draws for these tuning runs can be easily as high as for the estimation itself. My experience with the random walk algorithm is that it is quite impossible to tune the covariance to make all parameter sequences simultaneously look like they should. How should they look like?
Figure 14 shows three sequences and the associated autocorrelations for one parameter. The upper figure is the consequence of a too high variance. There are too few candidate draws being accepted. The lower graph shows a sequence with a too small variance. Both choices result in a slowly decreasing autocorrelation function as can be seen in the right graphs. The middle graph is a sequence with an appropriate variance choice. This eyeball test is the first hint of a wrong choice and after some experience eyeballing comes close to a calculated autocorrelation function.
Fernández-Villaverde and Rubio-Ramírez (2004a) check robustness by running several sequences with different start vectors. If these sequences and the trial runs to detect an appropriate innovation variance are run simultaneously and not sequentially, then one can assure robustness with respect to start values, calculate unbiased convergence diagnostic tests and in the proposed variant automatically arrives at an appropriate choice of the random walk shock variances.

Figure 14: Random Walk Covariance Choice

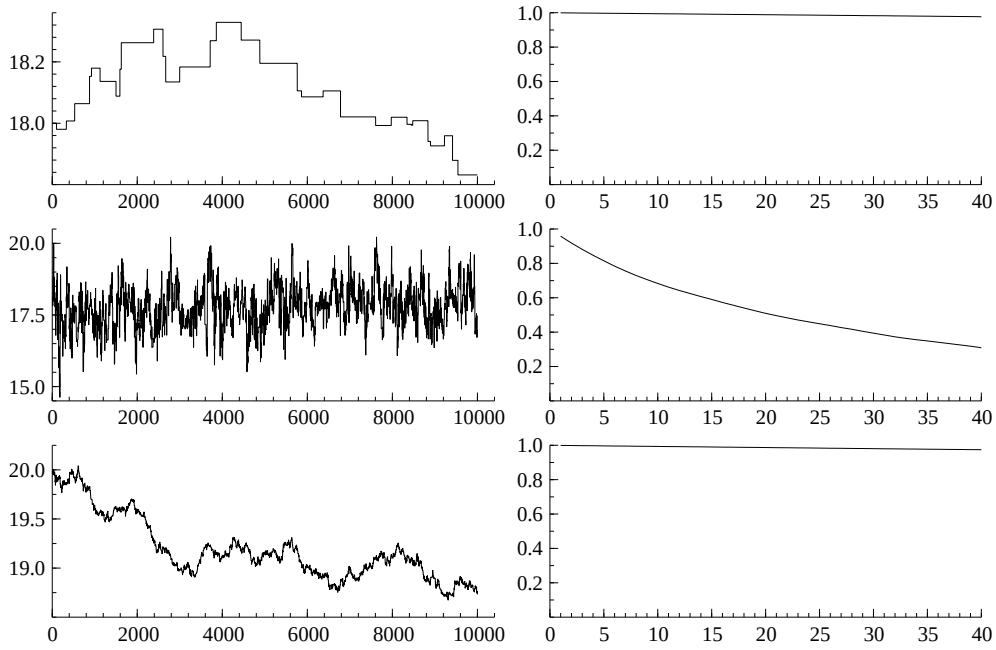


Table 4: Genetic Metropolis-Hastings Algorithm

1. choose $\hat{\boldsymbol{\theta}}_m$, $m = 1, ..., M$ and N
2. repeat the following steps starting with $n = 1$
   (a) repeat for $m = 1, ..., M$
      i. draw m_1 and m_2 such that $m_1 \neq m_2 \neq m$
      ii. generate $\hat{\boldsymbol{\theta}}_m^* = \hat{\boldsymbol{\theta}}_m + \gamma^{GE}(\hat{\boldsymbol{\theta}}_{m_1} - \hat{\boldsymbol{\theta}}_{m_2}) + \boldsymbol{\epsilon}$, where $\boldsymbol{\epsilon} \sim \mathcal{U}(-b, b)^D$
      iii. $\hat{\boldsymbol{\theta}}_m = \hat{\boldsymbol{\theta}}_m^*$ if $U(0,1) \leq \frac{p(\boldsymbol{y}_{1:t}^0 \mid \hat{\boldsymbol{\theta}}_i^*)\, p(\hat{\boldsymbol{\theta}}_i^*)}{p(\boldsymbol{y}_{1:t}^0 \mid \hat{\boldsymbol{\theta}}_i)\, p(\hat{\boldsymbol{\theta}}_i)}$
      iv. record $\hat{\boldsymbol{\theta}}_{n,m} = \hat{\boldsymbol{\theta}}_m$
   (b) calculate diagnostic test, set $J = n$ if $R(\hat{\boldsymbol{\theta}}_{1:n,1:M}) < 1.1$
   (c) if $n < N$, $n = n + 1$, goto 2(a)
3. disregard burn-in draws $\hat{\boldsymbol{\theta}}_{1:J,1:M}$

The draws are collected again in a $N \times D \times M$ tensor $\hat{\boldsymbol{\theta}}$ with M sequences of N draws for D parameters. A candidate draw n^* for a parameter vector in a sequence m_i is partly generated as in the random walk variant by a random shock added to the last parameter draw. Moreover, and this is new, the difference of two parameter vectors from two randomly chosen sequences m_1 and m_2 is added.

$$\hat{\boldsymbol{\theta}}_{n^*,m_i} = \hat{\boldsymbol{\theta}}_{n,m_i} + \gamma^{GE}(\hat{\boldsymbol{\theta}}_{n,m_1} - \hat{\boldsymbol{\theta}}_{n,m_2}) + \boldsymbol{\epsilon} \tag{47}$$

where $\boldsymbol{\epsilon} \sim U(-b, b)^D$ and $\gamma^{GE} = 2.38/\sqrt{2D}$ for a normal target density. I also tested zero mean normally distributed shocks without changing the results. The parameters γ^{GE} and the shock bounds determine the relative weight of cross and random innovations.
If the variance of the target density is $\boldsymbol{\Sigma} = \mathrm{Cov}(\boldsymbol{\theta})$ then the variance of the difference of two population parameter vectors from the sequences m_1 and m_2 is $\mathrm{E}[(\boldsymbol{\theta}_{m_1} - \boldsymbol{\theta}_{m_2})(\boldsymbol{\theta}_{m_1} - \boldsymbol{\theta}_{m_2})'] = 2\boldsymbol{\Sigma}$. In case of a converged sequence we get by the law of large numbers $\lim_{N\to\infty} \sum_{n=1}^{N} (\hat{\boldsymbol{\theta}}_{n,m_1} - \hat{\boldsymbol{\theta}}_{n,m_2})(\hat{\boldsymbol{\theta}}_{n,m_1} - \hat{\boldsymbol{\theta}}_{n,m_2})' = 2\boldsymbol{\Sigma}$. Therefore the optimal scale parameter for a normal distribution in this algorithm is $\gamma^{GE} = \gamma^{RW}/\sqrt{2}$.
The intuition behind this procedure is that the variance of the difference between two randomly drawn parameters is the optimal one given that the sequence has converged. The idea originates from the diagnosis test where the within and between sequence variance is examined and convergence is detected when they are the same. I have found the same construction to generate candidate vectors in a working paper by ter Braak (2004) from a biological institute. He argues that this way to generate candidate vectors is also used in the global genetic maximization algorithm called differential evolution by Storn and Price (1995).
The rest of the algorithm corresponds to the random walk variant. The candidate draw and the last draw are used to calculate the posterior ratio which together with a random number determines the acceptance of the candidate. Instead of D variances as in the random walk variant this algorithm has only two free parameters γ^{GE} and b to be tuned in trial runs.
This genetic extension allows a very simple parallelization of the code for an estimation on a cluster of computers. Each computer generates one sequence and the only information to be communicated between the computers is the $\hat{\boldsymbol{\theta}}_{1:M}$ matrix in step 2(a)iv. Its size is only $D \times M$ so that communication costs are mainly determined by the latency of the network. This procedure exhibits a very favorable linear acceleration since doubling the number of computers also doubles the estimation speed.
Section 7 presents a Monte Carlo simulation where the one sequential random walk Metropolis-Hastings is compared to the proposed genetic extension.

6 Marginal Likelihood

Model selection is a difficult but important matter and depends usually on a variety of more or less formal criteria and the given application. An frequent and rather informal approach is to examine the models ability to replicate some moments of the data. Another important procedure is to select alternative candidates by their out-of-sample record and Friedman (1953) wrote: '*The only relevant test of validity of a hypothesis is comparison of its predictions with experience.*'
The criterion derived in the following is a general likelihood based criterion and according to Berger and Wolpert (1988) the likelihood contains all relevant information needed. One challenge is that models of interest are often nonnested and do not emerge from each other through parameter restrictions so that classical likelihood ratio tests are not of much help. In practice functional forms, the number of estimated and calibrated parameters or the shock distributions may differ across alternative models.
Another problem is that models are inherently wrong since they are not a true representation but approximations of the reality and are designed to explain some features of the real world in a given application. This is a somewhat delicate statement within a classical approach which adheres to the notion of a true parameter or data generating process.
Landon-Lane (1998) discusses the Bayesian model selection within one-dimensional linear processes. The nonnested nature of alternative models as well as the fact that model are never a true representation of the world can be addressed within a Bayesian framework as described by Fernández-Villaverde and Rubio-Ramírez (2004a). Moreover they address the criticism of the Bayesian model selection to depend on the model priors. They show that asymptotically the best model under the Kullback-Leibler measure will have the highest posterior probability.
For some competing models $\{M_1, ..., M_m\}$, parameter priors and observable densities, the unobservables can be integrated out to arrive at the marginal likelihood

$$p(\boldsymbol{y} \mid M_i) = \int_{\Theta_{M_i}} p(\boldsymbol{y} \mid \boldsymbol{\theta}_{M_i}, M_i)\, p(\boldsymbol{\theta}_{M_i} \mid M_i)\, d\boldsymbol{\theta}_{M_i}. \tag{48}$$

The parameter posterior is used for the inference conditional on the adequacy of the model whereas the marginal likelihood is used for a criticism of the entertained model in the light of data.
For any two models (M_i, M_j) and their respective priors $p(M_i)$ and $p(M_j)$ the Bayes formula gives

$$\frac{p(M_i \mid \boldsymbol{y})}{p(M_j \mid \boldsymbol{y})} = \frac{p(M_i)}{p(M_j)} \frac{p(\boldsymbol{y} \mid M_i)}{p(\boldsymbol{y} \mid M_j)}.$$

The expression on the left hand side is the posterior odds ratio. On the right hand side the prior odds ratio is transformed by the Bayes factor. Again the

(marginal) likelihood or evidence transforms a prior density into a posterior. A posterior odds ratio greater than 1 favors model M_i and M_j otherwise. There is no confidence interval or significance level for this number.
The most difficult part is to calculate the marginal likelihoods which constitute the Bayes factor. Most of the work to calculate the marginal likelihood is already done once the Metropolis-Hastings algorithm has converged and generated parameter draws from the posterior density and the associated posterior values. Gelfand and Dey (1994) show that with any density $h(\boldsymbol{\theta}_{M_i} \mid M_i)$ we can write

$$\begin{aligned}
& \mathrm{E}_{p(\boldsymbol{\theta}_{M_i} \mid \boldsymbol{y}, M_i)} \left(\frac{h(\boldsymbol{\theta}_{M_i} \mid M_i)}{p(\boldsymbol{y} \mid \boldsymbol{\theta}_{M_i}, M_i)\, p(\boldsymbol{\theta}_{M_i} \mid M_i)} \right) \\
= & \int_{\Theta_{M_i}} \frac{h(\boldsymbol{\theta}_{M_i} \mid M_i)}{p(\boldsymbol{y} \mid \boldsymbol{\theta}_{M_i}, M_i)\, p(\boldsymbol{\theta}_{M_i} \mid M_i)} p(\boldsymbol{\theta}_{M_i} \mid \boldsymbol{y}, M_i)\, d\boldsymbol{\theta}_{M_i} \\
= & \int_{\Theta_{M_i}} \frac{h(\boldsymbol{\theta}_{M_i} \mid M_i)}{p(\boldsymbol{y} \mid \boldsymbol{\theta}_{M_i}, M_i)\, p(\boldsymbol{\theta}_{M_i} \mid M_i)} \frac{p(\boldsymbol{y} \mid \boldsymbol{\theta}_{M_i}, M_i)\, p(\boldsymbol{\theta}_{M_i} \mid M_i)}{\int_{\Theta_{M_i}} p(\boldsymbol{y} \mid \boldsymbol{\theta}_{M_i}, M_i)\, p(\boldsymbol{\theta}_{M_i} \mid M_i)\, d\boldsymbol{\theta}_{M_i}} d\boldsymbol{\theta}_{M_i} \\
= & \frac{\int_{\Theta_{M_i}} h(\boldsymbol{\theta}_{M_i} \mid M_i)\, d\boldsymbol{\theta}_{M_i}}{\int_{\Theta_{M_i}} p(\boldsymbol{y} \mid \boldsymbol{\theta}_{M_i}, M_i)\, p(\boldsymbol{\theta}_{M_i} \mid M_i)\, d\boldsymbol{\theta}_{M_i}} = p(\boldsymbol{y} \mid M_i)^{-1}.
\end{aligned}$$

According to the last equation all we have to do is to calculate a weighted mean of the Metropolis-Hastings sequence. Geweke (1999) proposes the following procedure. Calculate the mean and covariance of the parameter draws for each model M_i

$$\bar{\boldsymbol{\theta}}_{M_i} = \frac{1}{N} \sum_{n=1}^{N} \hat{\boldsymbol{\theta}}_{n,M_i} \qquad \hat{\boldsymbol{\Sigma}}_{M_i} = \frac{1}{N} \sum_{n=1}^{N} (\hat{\boldsymbol{\theta}}_{n,M_i} - \bar{\boldsymbol{\theta}}_{M_i})(\hat{\boldsymbol{\theta}}_{n,M_i} - \bar{\boldsymbol{\theta}}_{M_i})'.$$

If k_{M_i} denotes the number of estimated parameters of a model, define a χ^2 critical value for quantile p

$$\boldsymbol{\Theta}_{M_i} = \left\{ \boldsymbol{\theta} : (\boldsymbol{\theta} - \bar{\boldsymbol{\theta}}_{M_i})' \hat{\boldsymbol{\Sigma}}_{M_i}^{-1} (\boldsymbol{\theta} - \bar{\boldsymbol{\theta}}_{M_i}) \leq \chi^2_{1-p}(k_{M_i}) \right\}.$$

To assure robustness we should examine the results for different quantiles. Geweke (1999) proposes to use quantiles $p = 0.1, ..., 0.9$. The density $h(.)$ is specified to be

$$h(\boldsymbol{\theta}) = p^{-1} (2\pi)^{-\frac{k_{M_i}}{2}} \left| \hat{\boldsymbol{\Sigma}}_{M_i} \right|^{-\frac{1}{2}} \exp\left(-\frac{1}{2} (\boldsymbol{\theta} - \bar{\boldsymbol{\theta}}_{M_i})' \hat{\boldsymbol{\Sigma}}_{M_i}^{-1} (\boldsymbol{\theta} - \bar{\boldsymbol{\theta}}_{M_i}) \right) I_{\boldsymbol{\Theta}_{M_i}}(\boldsymbol{\theta})$$

with an indicator function $I_S(s) = 1$ if $s \in S$ and 0 otherwise. The estimator of the marginal likelihood is finally given by

$$\hat{p}(\boldsymbol{y} \mid M_i) = \left(\frac{1}{N} \sum_{n=1}^{N} \frac{h(\hat{\boldsymbol{\theta}}_{n,M_i})}{p(\boldsymbol{y}^0 \mid \hat{\boldsymbol{\theta}}_{n,M_i}, M_i)\, p(\hat{\boldsymbol{\theta}}_{n,M_i} \mid M_i)} \right)^{-1}.$$

7 Results

In this section the performance of the algorithms is presented. It is divided in two subsections - one for the solution and one for the estimation part. The calibrated parameters for the data simulations are the same as in Fernádez-Villaverde and Rubio-Ramírez (2004b) to allow a direct comparison of the results. Two different parameter calibrations are used. The first scenario is the benchmark case with an almost linear policy function. The second parameter vector represents the nonlinear scenario and implies a curved policy function.
The first topic in the solution part is the approximation accuracy of the linear and nonlinear solutions according to the Euler equation error. The second question is how much the Smolyak operator reduces the required computational effort to achieve a level of accuracy comparable to the approximation with the Kronecker operator.
The second part discusses the likelihood evaluation, the genetic Metropolis-Hastings performance, the parameter estimates and the marginal likelihoods. The likelihood evaluations of the Kalman, Gaussian, Gaussian particle and bootstrap filter are compared. The random walk and the proposed genetic Metropolis-Hastings algorithms are compared in a Monte Carlo simulation study. The parameter estimates are presented for both scenarios and the marginal likelihoods finally selects between the nonlinear and the linearized model estimates.

7.1 Policy

Table 5 shows the bounds of a flat prior and both scenarios represented by different parameter sets. The parametrization differ in risk aversion parameter τ and

Table 5: Calibrated parameters

parameters	θ	ρ	τ	α	β	δ	σ_e	σ_{ϵ_y}	σ_{ϵ_l}	σ_{ϵ_i}
benchmark	.357	.95	2	.4	.99	.02	.007	.000158	.0011	.000866
risky	.357	.95	50	.4	.99	.02	.035	.000158	.0011	.000866
upper prior	0	0	0	0	.75	0	0	0	0	0
lower prior	1	1	100	1	1	.05	.1	.01	.01	.01

standard deviation of the productivity shock σ_e. Risk aversion combined with large shocks implies a curved policy function incorporating a risk premium. The policy does not include the deterministic steady state and together with the curvature it induces relatively large errors in the solution derived from linearization. The main effect in the risky scenario is that all variables exhibit a higher variance since the driving productivity variance rises from 0.022 to 0.11.
Figure 15 shows the policies in the benchmark scenario derived from a linearization and the nonlinear solution. The consumption policies do not differ substantially whereas the labor policy is slightly curved even for the benchmark

Figure 15: Policy Functions in the Benchmark Scenario

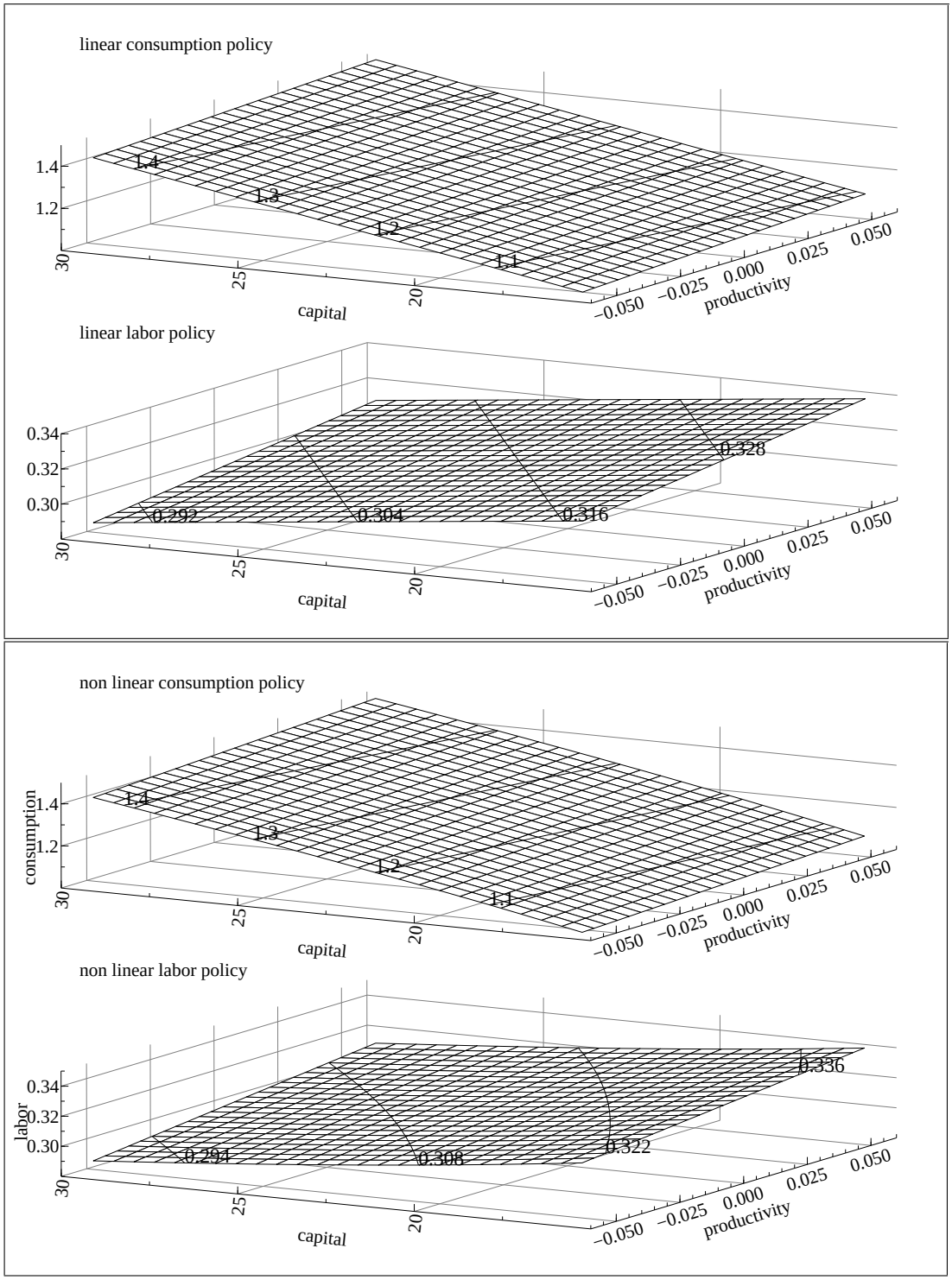


Figure 16: Euler Error in the Benchmark Scenario

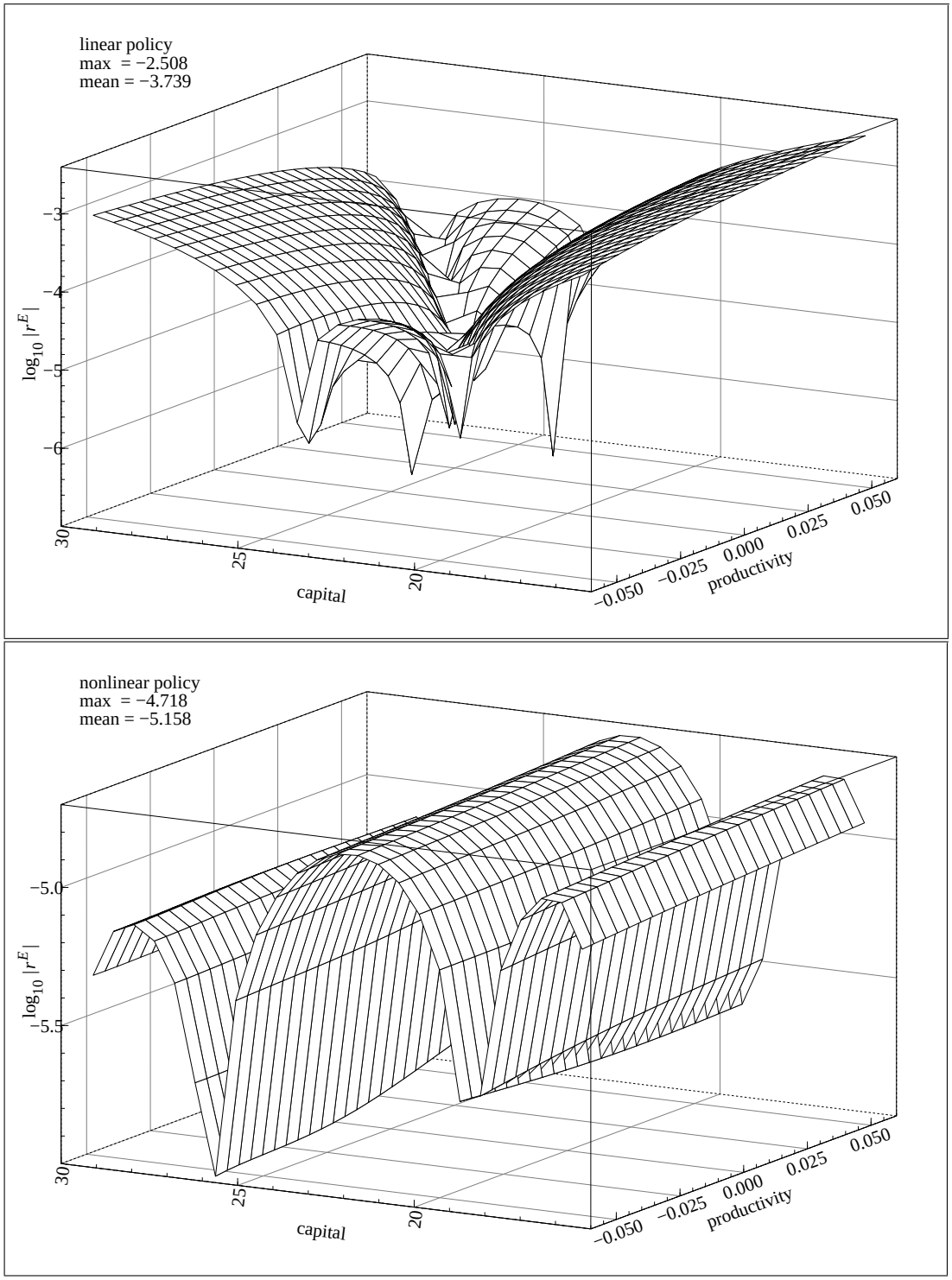


Figure 17: Policy Functions in the Risky Scenario

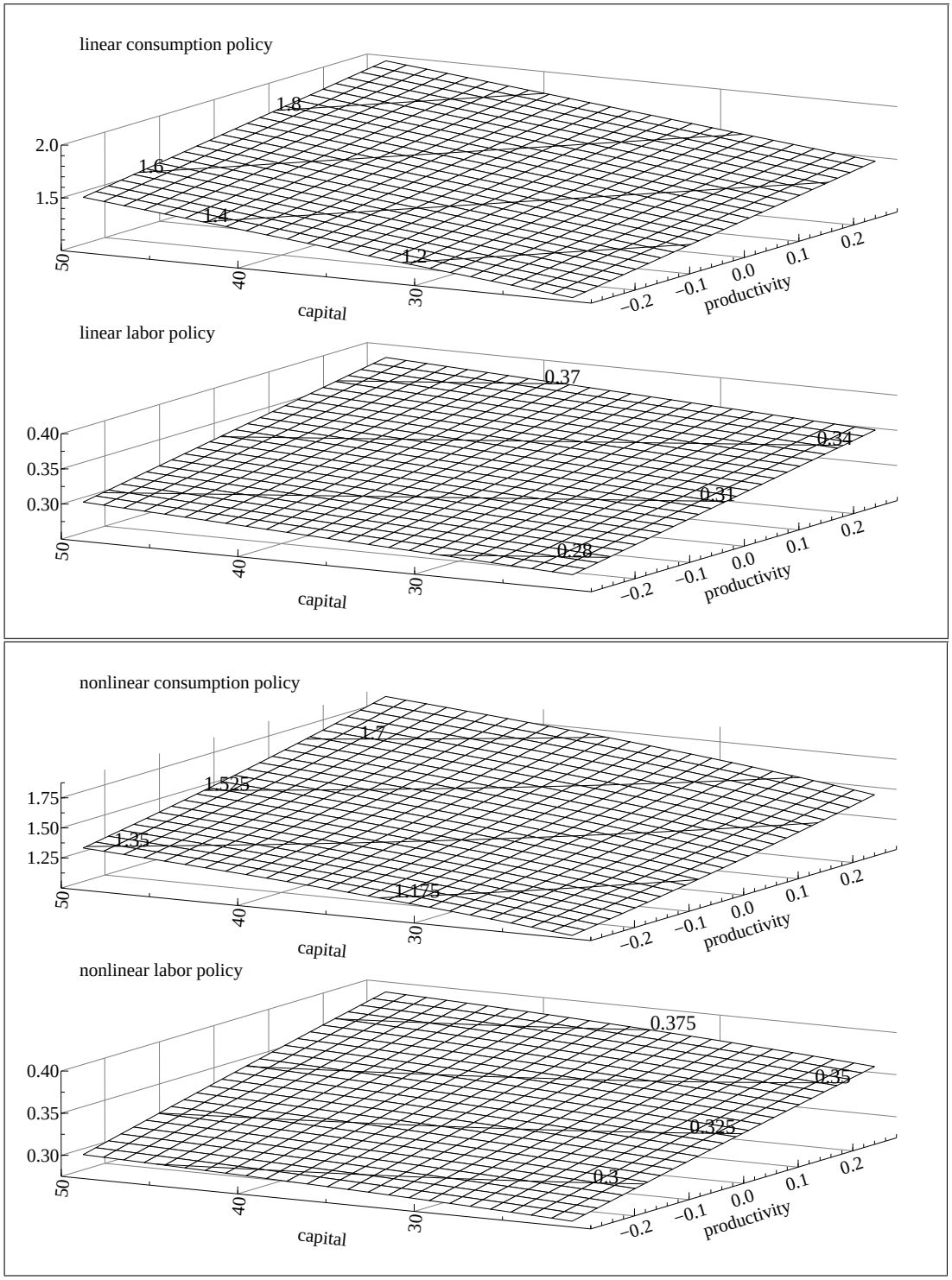


Figure 18: Euler Error in the Risky Scenario

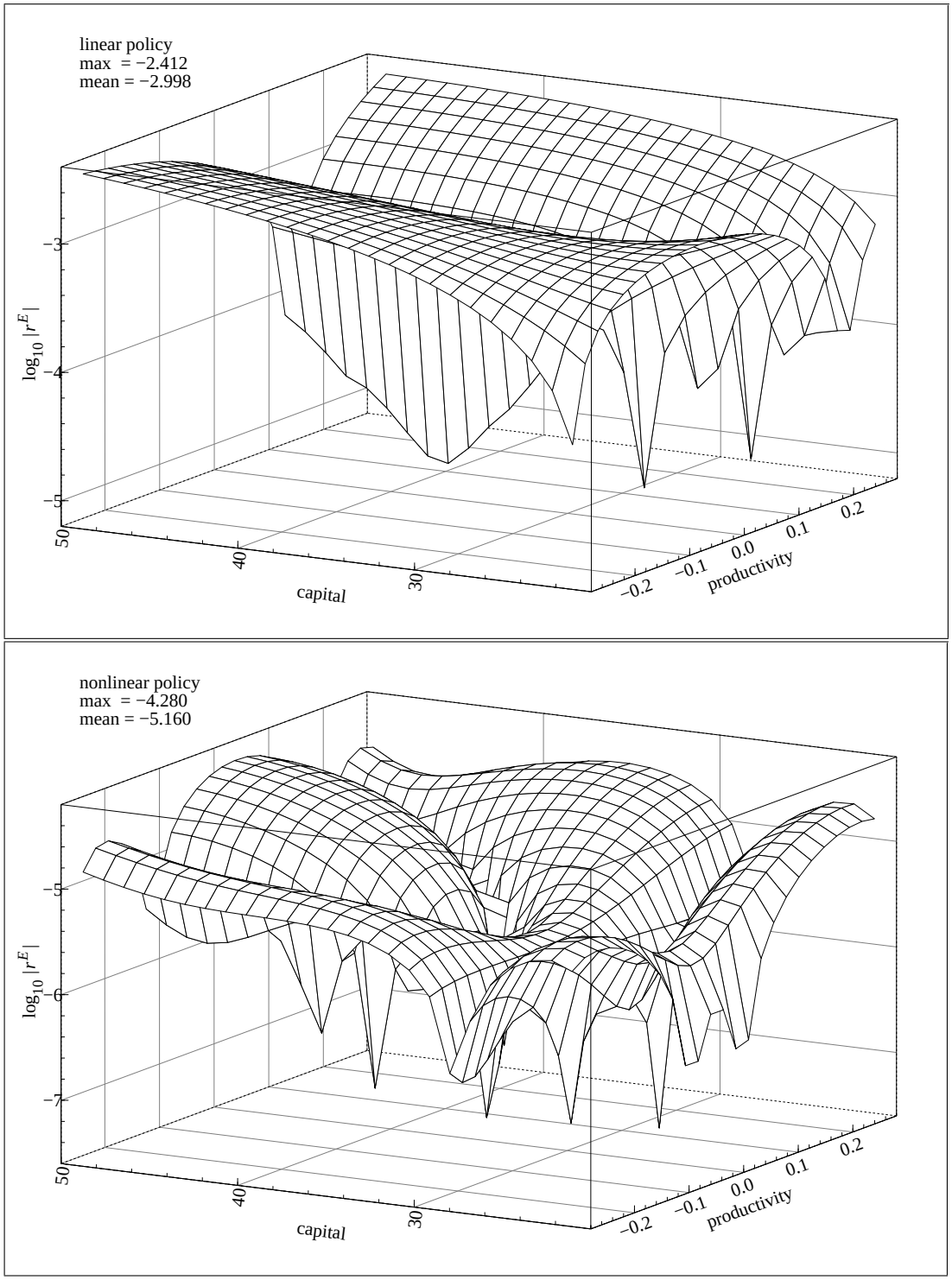


Figure 19: Smolyak Euler Error

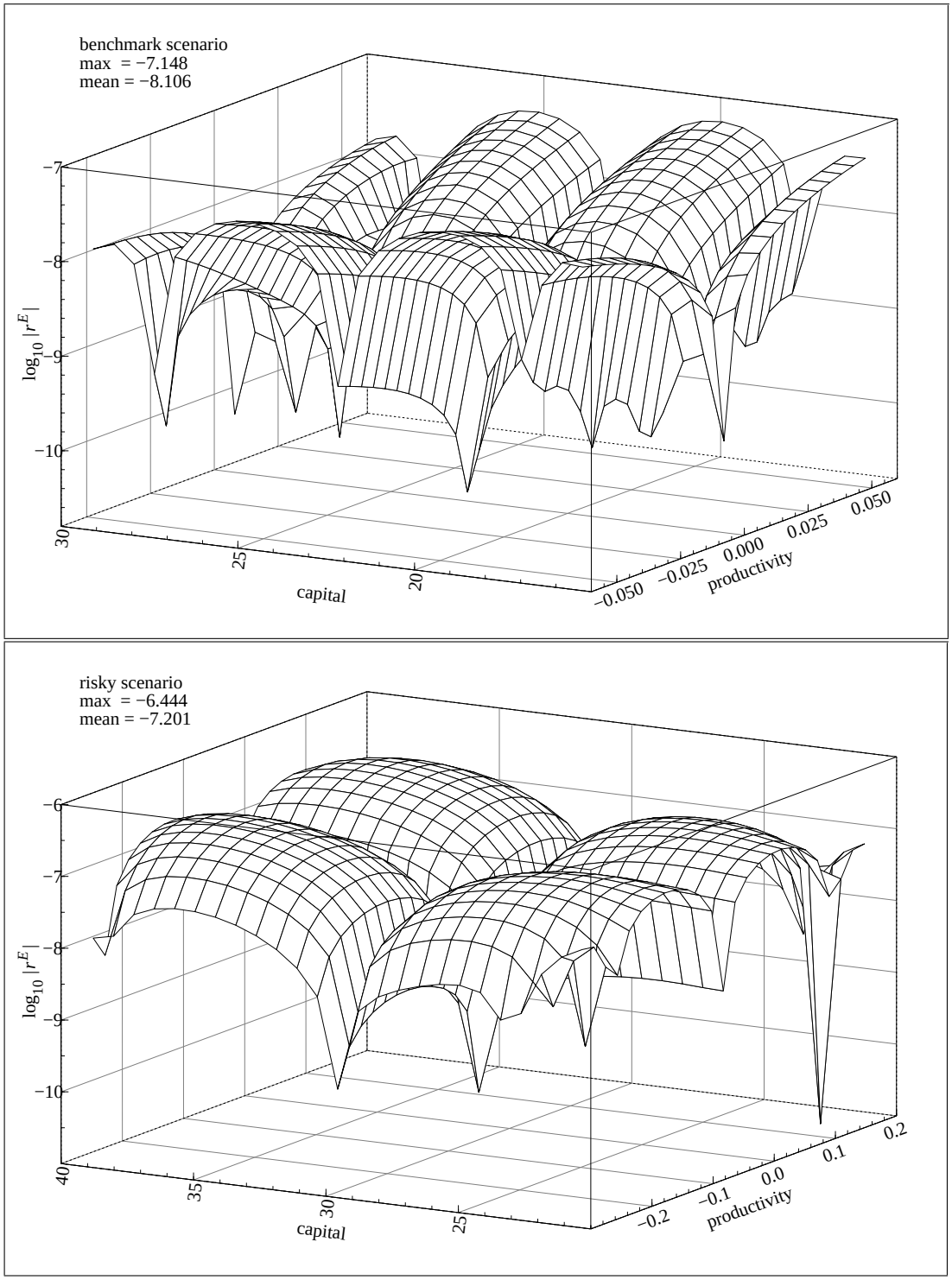
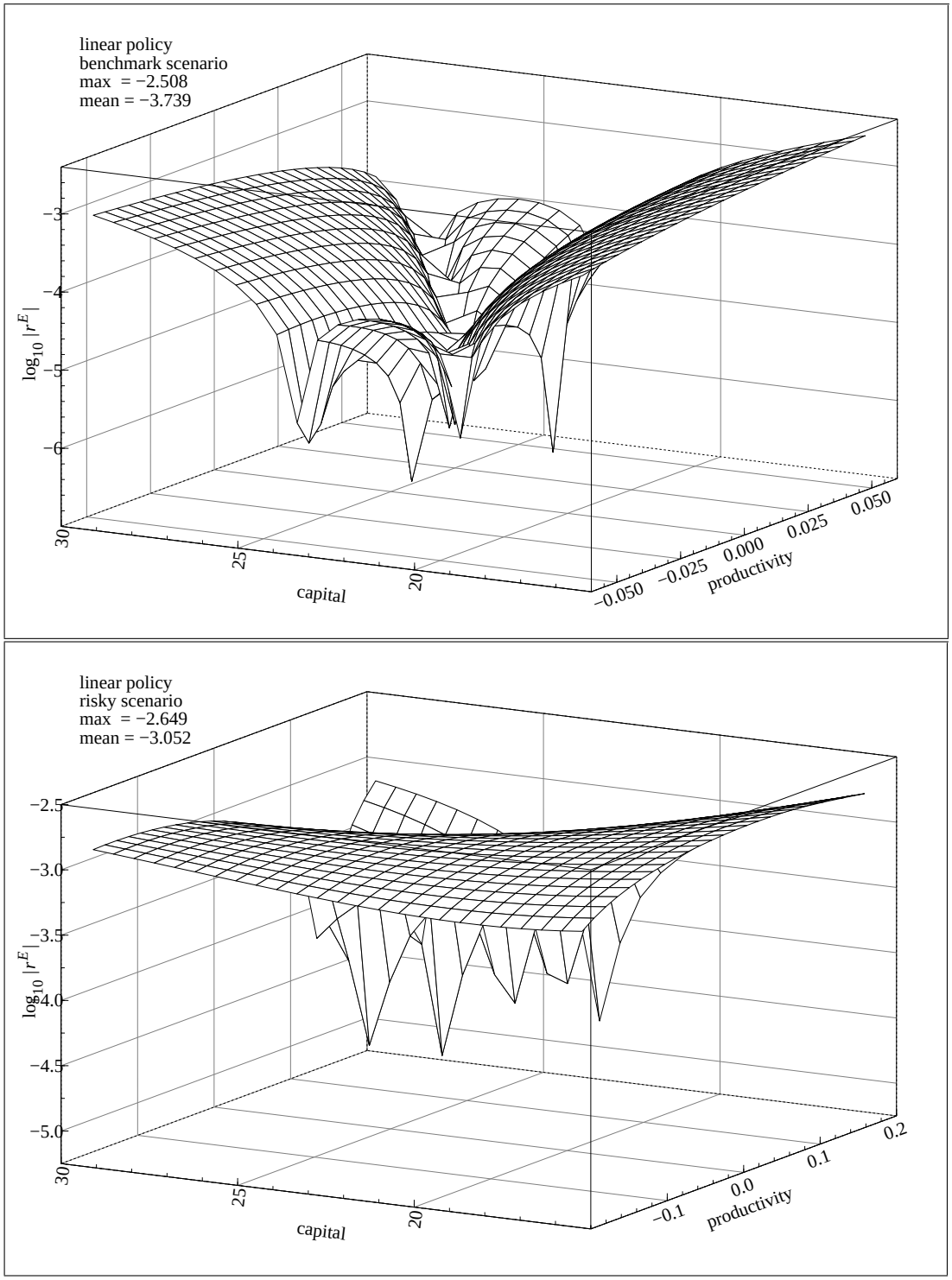


Figure 20: Euler Error of Linearization with Analytical Derivatives



parameter set. Whether these differences translate in a substantial estimation bias or not can hardly be judged by these plots alone. A first hint is to have a look on the implied Euler errors shown in figure 16. The linearized policy induces a maximal relative loss of around 1% of the consumption expenditures whereas the nonlinear policy maker suffers 100 times less from the use of the approximation instead of the exact optimal policy. As expected the linearized solution deteriorates when moving away from the steady state.
The solutions for the nonlinear scenario are given in figure 17. The absence of the certainty equivalence is represented in the Euler error of the linearized policy. They are larger around the steady state compared to the benchmark scenario.
The nonlinear policies were approximated by a 4^{th} degree Chebyshev polynomial on a 5×5 grid constructed by the Kronecker product. Figure 19 shows the Euler error for a 4^{th} level Smolyak approximation in both scenarios. In order to analyze the reduction of the computational effort by the Smolyak operator I calculated for the benchmark and risky scenario the nonlinear solutions with rising accuracy. The results are given in table 6. For the Smolyak operator the level q in equation

Table 6: Smolyak Reduction

Product Rule				Smolyak Operator			
Benchmark scenario							
Degree	Nodes	Max	Mean	Level	Nodes	Max	Mean
2	9	-3.81	-4.26	1	5	-3.22	-4.27
4	25	-5.51	-6.03	2	13	-5.13	-5.99
7	64	-7.96	-8.47	3	29	-7.15	-8.10
9	100	-9.47	-10.03	4	65	-9.97	-10.16
Risky scenario							
Degree	Nodes	Max	Mean	Level	Nodes	Max	Mean
1	4	-2.72	-3.07	1	5	-3.00	-4.28
3	16	-4.72	-5.64	2	13	-4.83	-5.86
5	36	-6.10	-7.12	3	29	-6.44	-7.20
8	81	-7.83	-8.70	4	65	-7.86	-8.35

(17) or (18) has to be chosen. The resulting number of nodes for the Kronecker product rule and the Smolyak operator are given in the column Nodes and the associated maximum and mean Euler error in the columns Max and Mean. In the benchmark parametrization the operator cuts the numbers of nodes by half to achieve approximately the same accuracy. In the risky scenario the dominance is lower and around 20%. The Smolyak reduction is lower in small dimensional problems and for larger models higher reduction in percentages can be expected. The finite element approach of Fernádez-Villaverde and Rubio-Ramírez (2004b) needs 140 nodes for a maximal Euler error of around -5, whereas the Smolyak approach achieves this number with only 13 nodes.

Linearization is done by numerical first derivatives. I checked the linearization approximation with analytical derivatives with the *Mathematica* perturbation code from Aruoba, Fernádez-Villaverde, and Rubio-Ramírez (2003). For the benchmark case the numerical solution gives an accuracy of 5 digits for the policy functions and 4 for the implied transition matrix. For the risky case I get a policy function accuracy of 5 digits and 3 for the state transition. The implications for the Euler error can be seen in figure 20. In the benchmark scenario the use of the numerical derivatives has neither a visible nor measurable effect. In the risky scenario the numerical derivatives increase the maximal error from -2.649 to -2.412 and the mean error from -3.052 to -2.998.

7.2 Estimates

This subsection presents the estimation results. In the likelihood evaluation section the approximation quality of the filters are of central interest. Then the proposed Metropolis-Hastings algorithm is tested against the random walk variant in a Monte Carlo simulation study. Afterwards I present the structural parameter estimation with the genetic Metropolis-Hastings algorithm and the Gaussian filter. The last calculations compare the linearized and the nonlinear estimation according to their marginal likelihoods on a nonlinearly generated data set.

7.2.1 Likelihood

The likelihood values at the calibrated parameters in table 7 give a first idea of the approximation quality of the filters discussed in chapter 4. Both Gaussian filters

Table 7: Log Likelihood Values

Filter	Benchmark Scenario	Risky Scenario
Kalman filter	1,365.12	-150,575.32
Gaussian Kalman filter	1,368.23	1,209.91
Gaussian particle filter	1,368.19	1,211.06
bootstrap filter	1,369.92	1,215.68

are calculated with level 3 Smolyak quadrature for the time and the measurement update. The Gaussian particle filter uses 5,000 and the bootstrap filter 40,000 particles. The Kalman likelihoods are identical for the linearization with analytical and numerical derivatives. In the benchmark case the bootstrap filter gives virtually the same likelihood as the Gaussian filter. The importance sampling step in the Gaussian particle filter hardly improves the likelihood accuracy in the risky and not at all in the benchmark scenario. The Kalman likelihood is zero for the risky scenario whereas Fernádez-Villaverde and Rubio-Ramírez (2004a) report a log value above 1000. For the benchmark scenario Fernádez-Villaverde

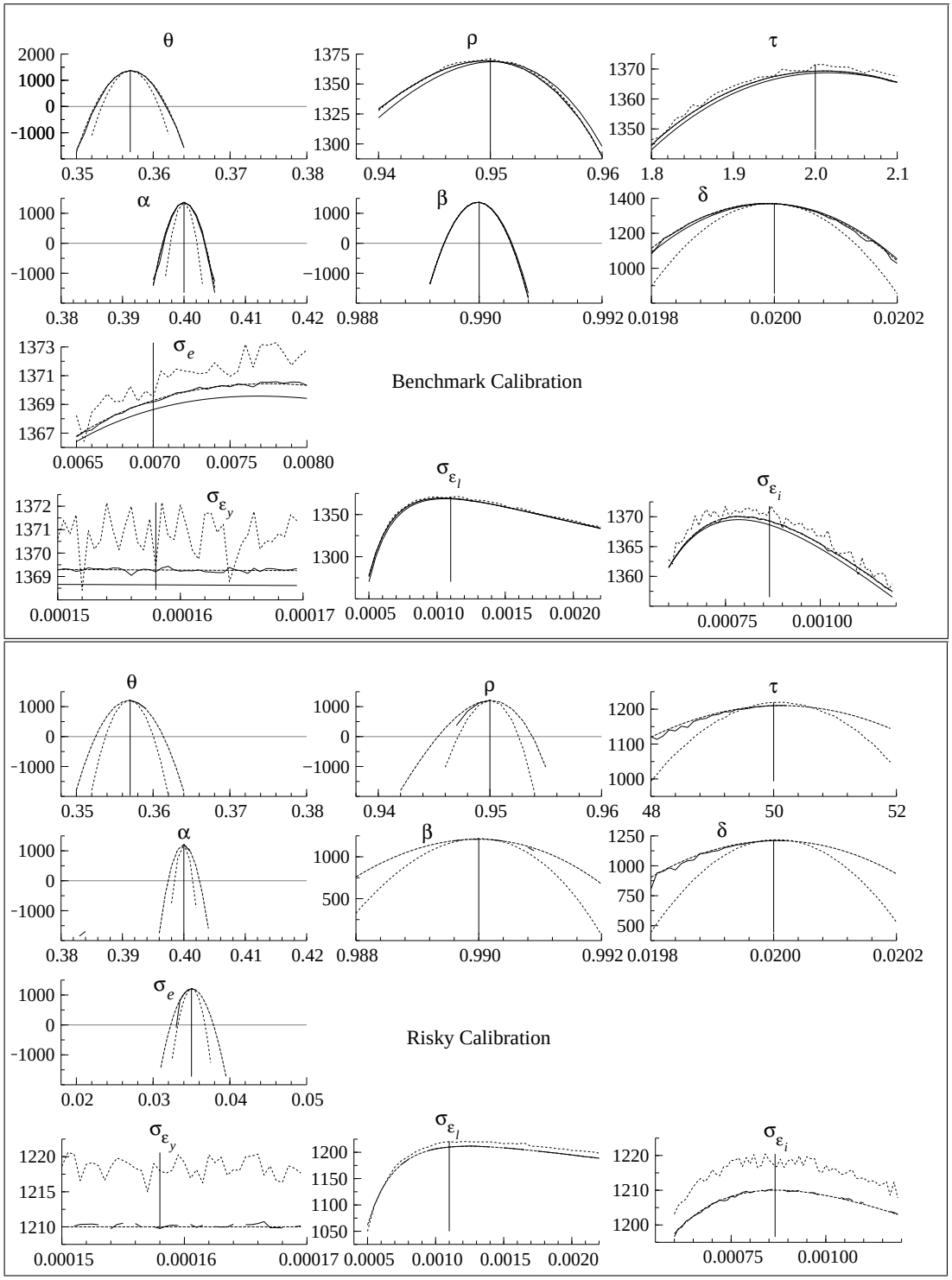
and Rubio-Ramírez (2004a) report 1462 and around 1000 for the bootstrap and Kalman likelihood, respectively.
Table 8 shows the convergence of the bootstrap filter when the number of particles is increased. The calculations are done with 50 replications. My results show a

Table 8: Convergence of the Bootstrap Filter

	Benchmark Scenario		Risky Scenario	
N	Mean	s.d.	Mean	s.d
10,000	1367.7	1.44	1213.5	9.9915
20,000	1368.0	0.74	1217.6	2.6814
30,000	1368.0	0.69	1218.0	1.6499
40,000	1368.1	0.62	1218.4	1.4650

lower standard deviation for the benchmark scenario whereas Fernádez-Villaverde and Rubio-Ramírez (2004b) report it the other way round.
The likelihood function is 10 dimensional and transversal cuts can be visualized by changing one parameter while keeping all others at their calibrated values. Figure 21 shows likelihood values for the filters in the benchmark and the risky scenario. The dotted graphs belong to the Kalman filter, the varying graph to the particle filter. Both Gaussian filters produce virtually the same graphs without any detectable differences. The likelihood around the true autoregressive parameter ρ is rather flat and the estimation procedure will have problems to find the true value. It is also not informative for risk aversion parameter τ and shock variances σ_e, σ_{ϵ_l}, σ_{ϵ_i} and σ_{ϵ_y}. Parameters θ, α, β, δ show a strongly peaked likelihood and good estimates can be expected.
The likelihood values of the Gaussian filters are similar to the bootstrap filter values. For the Gaussian filter I used a third level Smolyak quadrature. The time update quadrature uses 39 nodes for the three dimensional integration over both states and the state shock. The measurement update needs 151 nodes for the 5 dimensional integrals of states and measurement shocks. The spectral interpolation of the policy function in the next period for each particle or node is the most demanding computation. The Gaussian filter is around 100 times faster then the bootstrap filter. This dramatic reduction is of course due to the reduction of 40,000 particles to 190 quadrature nodes.
The likelihood traversal cuts for the risky calibration are shown in the lower part of figure 21. The Kalman filter evaluations are not plotted since they completely miss the true parameters. The likelihood values of all three nonlinear filters are again very similar. Fernádez-Villaverde and Rubio-Ramírez (2004b) report values around 830 whereas I calculated much higher values. There are differences between the benchmark and the risky scenario concerning the flatness of the likelihood. In the risky scenario the output and investment measurement shock variances are clearly peaked, indicating a more informative likelihood. Good

Figure 21: Transversal Cuts Through the Likelihood



estimates can be expected for autoregressive parameter ρ, risk aversion τ and productivity shock σ_e.
Due to the similarity of the likelihood values for the Gaussian filter and the bootstrap filter I will only present estimates obtained by the simple Gaussian filter.

7.2.2 Normal Density

This section presents a Monte Carlo simulation study where the random walk Metropolis-Hastings (RWMH) algorithm is compared with the proposed genetic extension (GEMH).
The known target density is a ten dimensional multivariate normal distribution with the benchmark parameters as the expected values and the reported standard deviations of Fernádez-Villaverde and Rubio-Ramírez (2004b) as variances. The covariances are zero. Taking the standard deviations as variances is necessary because the squared estimated standard deviations result in a singular covariance matrix with a determinant of 3.8E-99.
There are two questions I investigate. How fast do the algorithms converge? And how good are the random numbers generated after convergence? For this purpose I run 1,000 Metropolis-Hastings density estimations with the burn-in length detected automatically by a multiple reduction factor below 1.1. After the sequences have converged 50,000 subsequent draws are generated. The RWMH is run in three variants. The first (RWMHt) uses the scaled true target variance as the random walk variance. The other two (RWMHp) use the true variances multiplied with a small and a large random factor. The small factor is between 10^{-1} and 10^{1} and the large factor between 10^{-2} and 10^{2}. The GEMH is run with twenty parallel sequences. The innovation shock bound is $b = 10^{-5}$ and the parameter mixing factor is $\gamma^{GE} = 2.38/\sqrt{2 \times 10}$. Both number result in an acceptance ration of around 0.26. In a real application the true variances are not known and the RWMHt simulations are too optimistic and represent the optimal but not obtainable estimator quality. The perturbed variances correspond to the realistic situation of unknown variances.
The upper part of figure 22 plots the optimal RWMHt sequences. The solid straight lines represent the true parameter values and the dotted lines indicate a moving average. After some hundred draws the RWMHt sequence converge to the true values and after around 6,000 draws the diagnostic test detects convergence of the density estimate. The right hand draws after the burn-in sequence can be taken as a sample from the target density. The lower part of figure 22 shows the RWMHp sequences with the little perturbed variances. The convergence towards the true values is as fast as for the RWMHt but convergence of the density is detected only after 10,000 draws. Figure 23 shows one of the parallel sequences of the GEMH algorithm. Here the convergence towards the true value is faster if one counts only the draws of one sequence. This reflects the learning

Figure 22: Random Walk Metropolis-Hastings

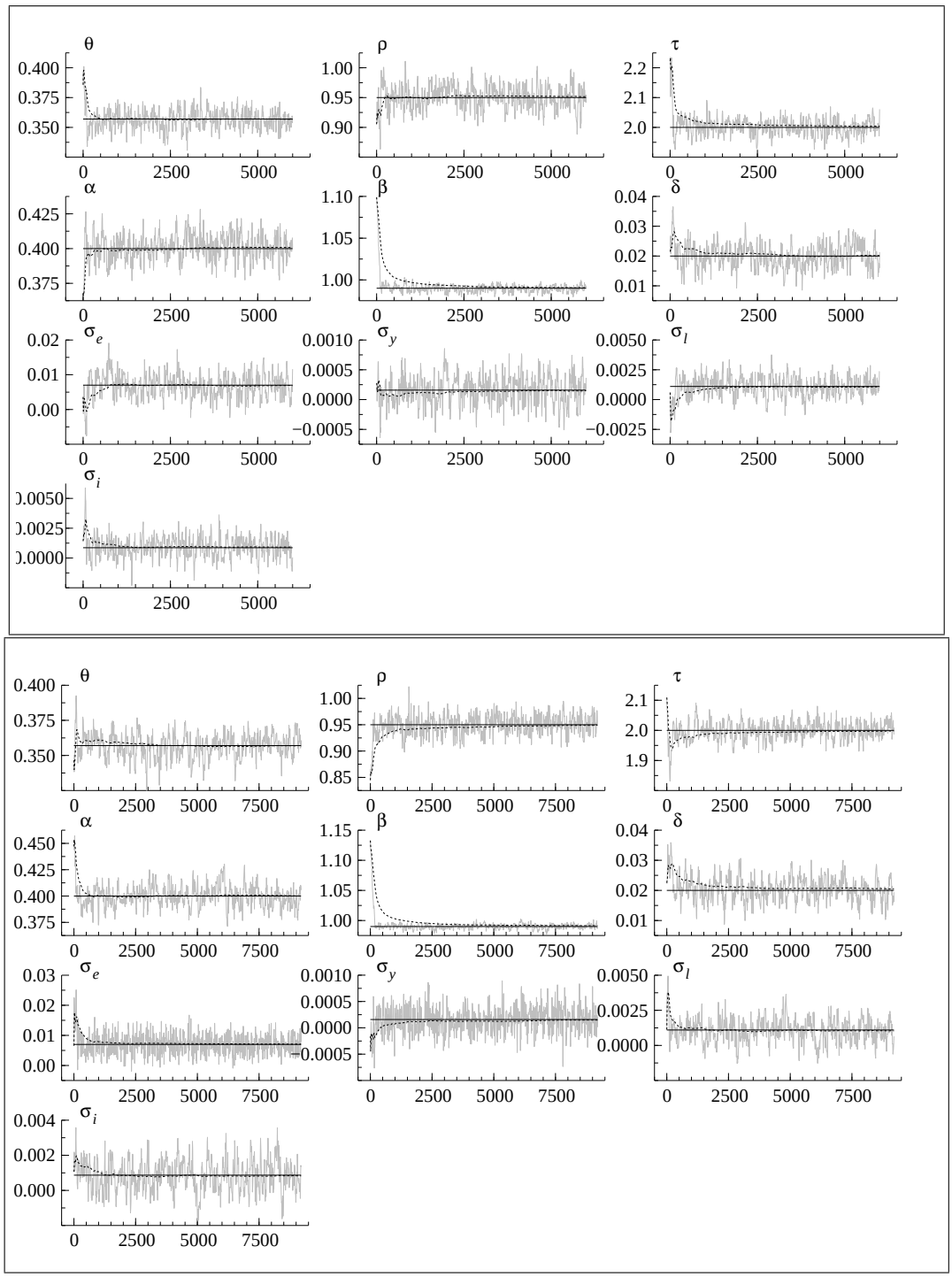
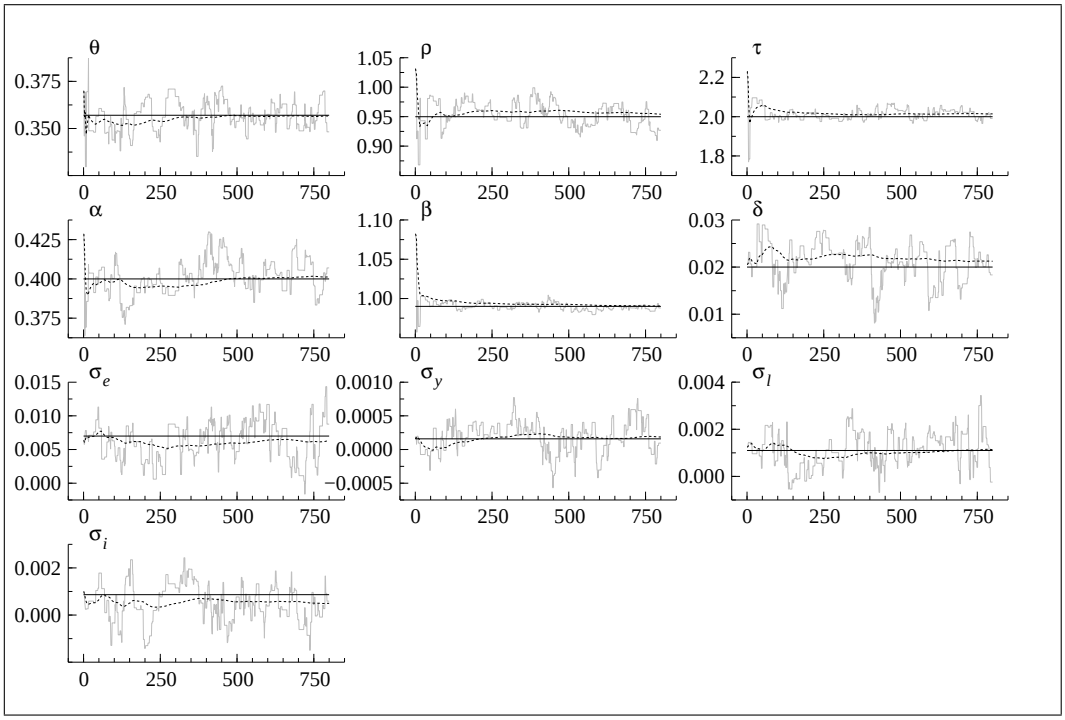


Figure 23: Genetic Metropolis-Hastings



process effectuated by the mixing of parallel sequences as an addition source of innovation beside the pure random walk shocks. This learning procedure has to be done manually when several trial sequences are run sequentially and not simultaneously.

Table 9 shows the summary of the simulation results. In the upper part the length of the burn-in sequences are compared. It also shows two different simulations with regard to the dispersion of the start values around the true values. The burn-in length of the GEMH is the total number of draws in all parallel sequences.

The burn-in length of the RWMHp depends mainly on the start value dispersion and the variance perturbation does not influence it. Not surprisingly, the RWMHt shows the fastest convergence. The RWMHp algorithm with the little perturbed variances is comparable to the RWMHt performance. The burn-in length of the GEMH is independent of the start value dispersion.

Whether the GEMH is competitive with the RWMHp or not depends on ones view about a representative deviation of the tuned variances from the true ones. To me it seems likely to use variances as far from the true ones as in the runs with a large factor perturbation. There is no sense in running trial sequences until the random shock variances represent the true ones since this estimate is the purpose of a final run. Or put it differently, the RWMHp sequences converge faster if the variances are tuned closed to the true ones in many trial runs or the

Table 9: Metropolis-Hastings Performance

		RWMHp		RWMHt	GEMH
		small	large		
start value dispersion	Burn-in Lengths				
narrow	max	26,400	104,200	15,000	20,000
	top .95	11,800	27,200	8,800	16,000
	mean	6,944	13,651	5,215	13,720
	low .05	3,000	4,800	2,600	12,000
	min	1,200	2,400	800	8,000
wide	max	22,000	55,200	17,000	20,000
	top .95	12,400	28,600	10,400	16,000
	mean	7,332	14,423	6,343	14,144
	low .05	3,400	5,400	3,200	12,000
	min	1,400	2,600	1,600	8,000
Random Number Quality					
μ					
	mean bias	1.01	1.85	1.00	0.96
	variance	1.38	3.70	1.00	1.23
σ					
	mean bias	8.95	31.05	1.00	4.96
	variance	1.42	3.39	1.00	1.34

sequences converge as slow as the GEMH due to wrong variances tuned in only a few runs.

If the performance of the highly perturbed RWMHp is taken as a realistic competitor for the GEMH then the result is that both algorithms achieve the same mean burn-in length. The difference between both algorithms is that in the trial sequences for the GEMH only two parameters have to be tuned. Moreover both parameters are the same for estimations independent of the size of the true variances. The RWMHp scale factor γ^{RW} needs some adjustment according to the size of the variance perturbation and the true variances.

The mean burn-in length is virtually the same for the RWMHp and GEMH. The variance of the GEMH burn-in length is much smaller and the RWMHp algorithm suffers from large outliers with the maximum as high as 104,200 draws. On the other side some estimations by the RWMHp algorithm converged much faster with a minimum burn-in length of 2,400. The result of this Monte Carlo simulation is that with respect to the burn-in length the GEMH is roughly comparable to the RWMHp. On average they converge after the same number of draws but the GEMH does not need trial sequences to find suitable random walk

shock variances.
The lower part of table 9 shows the random number quality. The mean of the 50,000 draws after the burn-in sequences is the estimated expected value. For each of the 1,000 simulations I calculated the absolute deviation of this estimate from the true value. The mean of these deviations is the mean bias which together with the variance characterizes the estimation quality. These measures are calculated for the expected values and the variances of the multinormal distribution. Then I divide these numbers by the corresponding optimal performance of the RWMHt simulations and take the mean over all 10 parameters.
The mean estimates of the small perturbed RWMHp and GEMH are comparable with the RWMHt estimates. The highly perturbed RWMHp shows the weakest performance. This also holds for the variance of the mean estimates. Estimates of the variances are not as good as the mean estimates for all algorithms but again GEMH performs better than RWMHp.

7.2.3 Posterior

This section reports the estimates of the posterior density of structural parameters of the model at hand from a sample with 100 observations. Two data sets are simulated - one with the benchmark and one with the risky parameter values. The parameters are calibrated for quarterly data and a sample represents 25 years. Both data sets are generated with the nonlinear solution. All posterior estimates are obtained by the Gaussian filter as the nonlinear estimator. As the linear estimator I take a linear perturbation with numerical derivatives and the Kalman filter.
The convergence test of the genetic Metropolis-Hastings algorithm consist of the acceptance ratios, the multivariate scale reduction factor R and the autocorrelation functions. The acceptance ratios have to be around .3 for each parallel sequence. The autocorrelation functions for each sequence have to fall fast to around .5 within 40 lags. They are very similar across sequences and parameters and I report only one representative autocorrelation function. The scale reduction factor R must fall below 1.1. The converged parallel sequences are stacked and a histogram represents the density estimate of the posterior.
One important decision is how to choose the start values for the sequences. Two different steps are necessary to obtain a posterior estimate. The first step has to find the modes and the second step samples around them to generate representative draws. For the maximization step the optimal mixing parameter and the shock variances are larger than for the sampling draws. I used the resampling step of the bootstrap filter for a posterior based mixing of the parallel parameter vectors. It accelerates the maximization substantially. The maximization procedure finds the mode within some thousand likelihood evaluations. The mode seems to be unique for the simple model. For multimodal densities specialized mixing strategies are available, see ter Braak (2004).

Table 10: Parameter Estimates

	True	mean		std.dev.	
		Nonlinear	Linear	Nonlinear	Linear
	Benchmark Scenario				
θ	0.35	0.358757	0.358176	1.83E-03	1.63E-03
ρ	0.95	0.921978	0.931711	1.90E-02	1.35E-02
τ	2.00	2.873962	2.637364	4.89E-01	3.75E-01
α	0.40	0.404878	0.403540	4.75E-03	4.20E-03
β	0.99	0.989154	0.989405	8.90E-04	7.71E-04
δ	0.02	0.021091	0.020784	1.06E-03	9.25E-04
σ_a	0.007	0.006349	0.006242	4.71E-04	3.92E-04
σ_y	0.000158	0.000672	0.000719	3.58E-04	3.34E-04
σ_l	0.0011	0.001271	0.001272	9.00E-05	8.00E-05
σ_i	0.000866	0.000592	0.000570	2.57E-04	2.50E-04
	Risky Scenario				
θ	0.35	0.356555	0.359832	4.47E-04	3.56E-03
ρ	0.95	0.950678	0.942740	8.57E-04	6.74E-03
τ	50.00	49.385802	62.910866	9.01E-01	2.07E+01
α	0.40	0.399076	0.408750	1.25E-03	9.20E-03
β	0.99	0.990152	0.988453	2.09E-04	1.79E-03
δ	0.02	0.019809	0.025163	2.62E-04	2.26E-03
σ_a	0.035	0.035053	0.031274	1.17E-04	2.11E-03
σ_y	0.000158	0.000160	0.000674	6.10E-06	4.58E-04
σ_l	0.0011	0.001080	0.001351	2.55E-05	9.24E-05
σ_i	0.000866	0.000877	0.000994	1.89E-05	2.15E-04

I do not present the maximization sequences but only the sampling step with start values narrowly dispersed around the true parameters. Starting from any parameter vector needs only more draws in addition to the burn-in draws thereafter without changing the density estimate. However, generating posterior histograms needs many more draws than finding the mode of the density.
Figure 24 present the diagnostics for the benchmark and the risky scenario estimation. The histograms are drawn for the obtained posterior values after the burn-in sequences. The vertical black lines indicate the last values of the parallel sequences. For both data sets the sampling takes place within some units of the log posterior. The acceptance ratio is around .3 and a representative autocorrelation function is falling fast. The scale reduction factor R is below 1.1 and indicates convergence of the density estimates.
The optimal parameters γ^{GE} and b were found to be independent of the estimator since the ones tuned in the linear estimation of the benchmark scenario were also optimal for the nonlinear estimations. I used the optimal mixing factor

Figure 24: Convergence Diagnostics

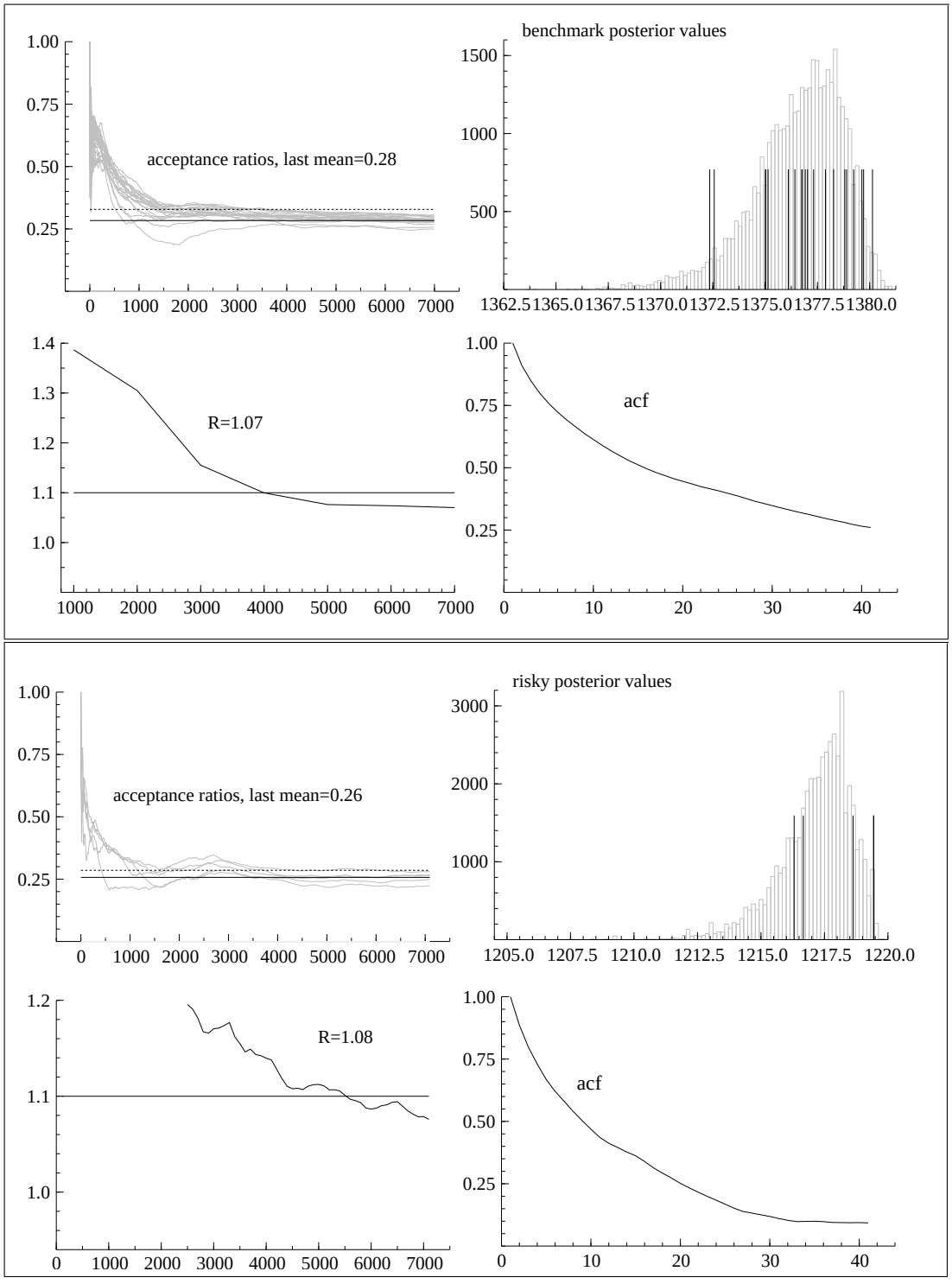


Figure 25: Parallel Metropolis-Hastings Burn-In Sequences

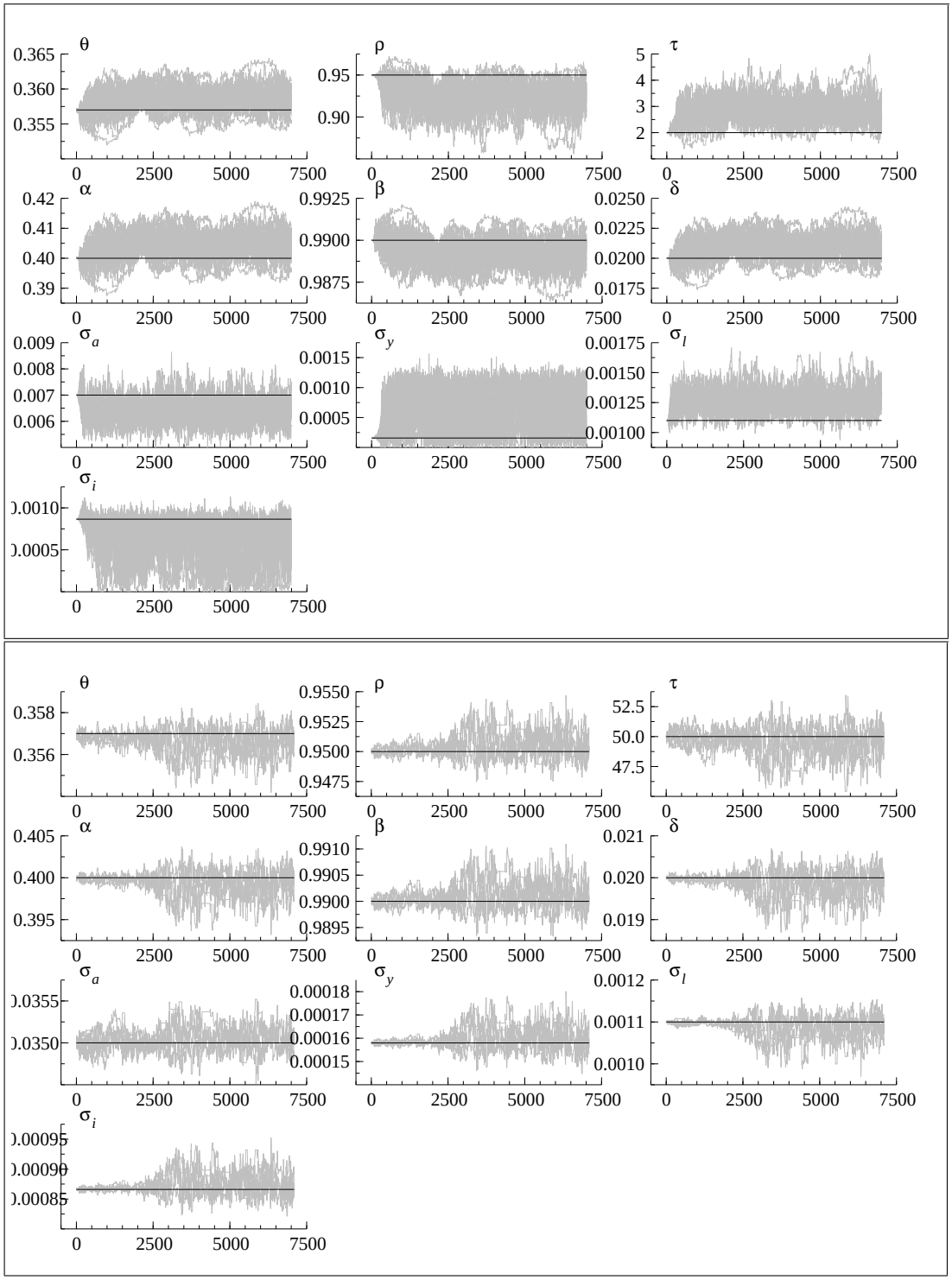
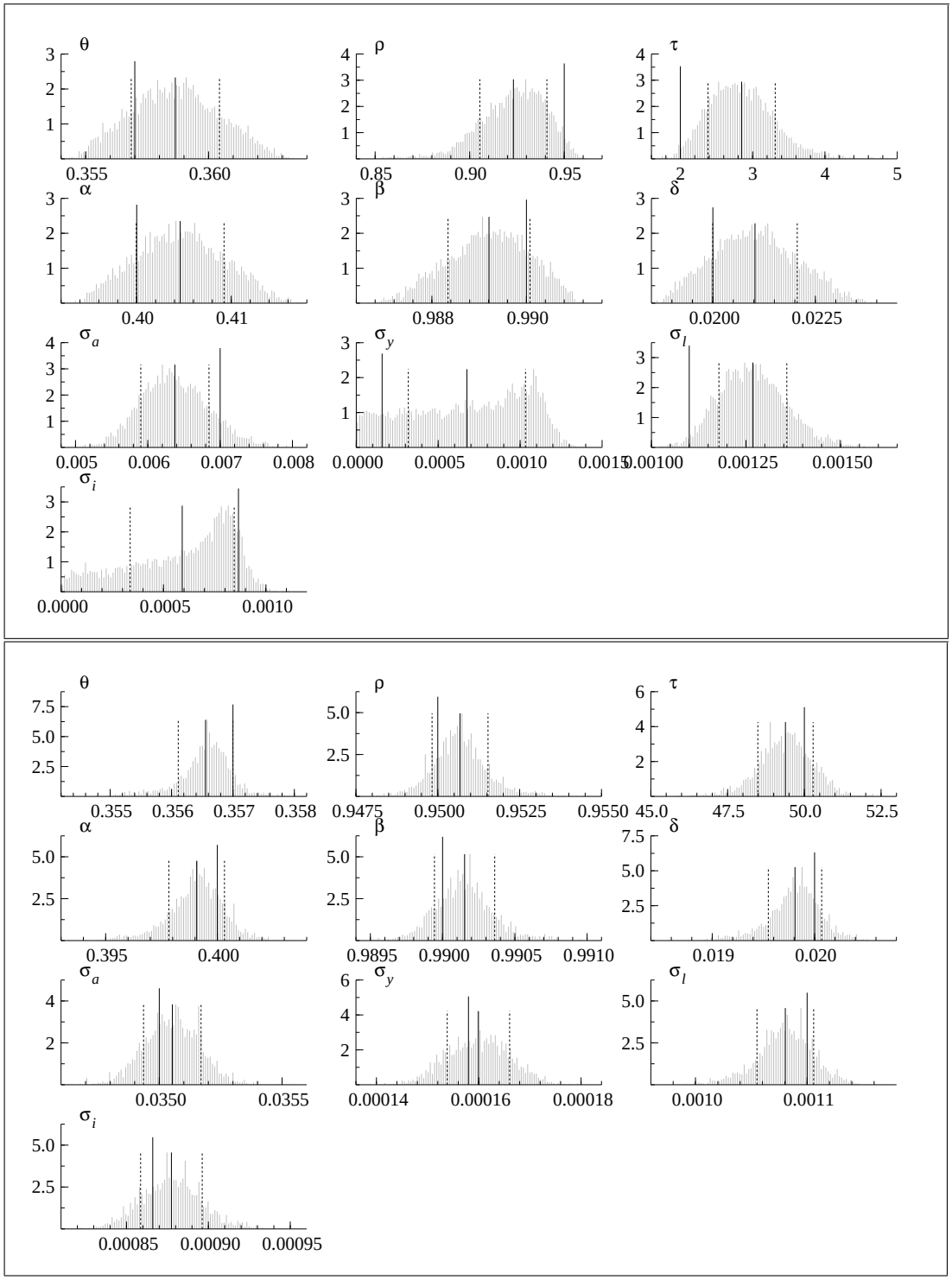


Figure 26: Estimated Posterior Densities



$\gamma^{GE} = 2.38/\sqrt{2 \times 10}$ scaled by 0.6 and a uniform shock bounds of $b = 10^{-9}$. The linear tuning runs took only some minutes and simplified the estimation process substantially since expensive nonlinear runs were superfluous. Whether this independence of the algorithm parameters from the estimator carries over to other models is to be tested. However, there is not much scope for tuning the parameters since γ^{GE} has an strong effect on the acceptance ratio. Changing the optimal mixing factor by a factor between 0.2 and 1 changed the acceptance ratio dramatically. The variance parameter of the random shocks b is of minor importance without to much influence on the acceptance ratio since most of the innovation comes from the mixing of the parallel sequences. In fact the number of parallel sequences is another but apparently less important parameter. It should depend on the number of parameters to be estimated and the number of modes of their posterior density. I have run the model estimations with 20 and 5 parallel sequences for the different parameter sets without noticeable implications.
In the benchmark case the potential scale reduction factor R fell below 1.1 after 5,000 draws from 20 sequences and in the risky case 7,000 draws from 5 sequences were needed. Afterwards the sequences were run until 50,000 posterior draws were generated.
Figure 25 shows all parallel burn-in sequences for the nonlinear estimations. The black horizontal lines represent the true parameters. None of the sequences is trending anymore and taken each separately they look similar to the middle graph in figure 14. The linear Kalman filter sequences are not shown. They move away from the true parameters towards their biased estimates.
Figure 26 shows the nonlinearly estimated parameter posteriors. The long black vertical line indicates the true values, the shorter lines the mean and the dotted lines bound two standard deviations around the mean. Table 10 presents the first moments of the posterior as point estimates. The uncertainty of these estimates is given by the second moment of the posterior. The Kalman filter estimates of the linearized model are listed as a comparison.
The benchmark parameter estimates are similar for the linear and nonlinear estimation. The standard deviations of the linear estimates are slightly smaller but the not maximum of the likelihood. The estimates of risk parameter τ, autocorrelation ρ and the shock variances are biased and unprecise. This is different to Fernádez-Villaverde and Rubio-Ramírez (2004b) who report better nonlinear estimates. This may be the driven by the sloppy solution of the singularity problem by measurement errors, a bug in the code or the simplicity of the Gaussian filter. A bug in the code and the Gaussian filter are unlikely the cause of the bias since the risky parameter point estimates are good and in line with Fernádez-Villaverde and Rubio-Ramírez (2004b). Moreover the Gaussian filter gives at least at the true benchmark parameters the same likelihood value as the bootstrap filter. Further experience with the code, other models and filters should resolve this puzzle.
The risky parameter point estimates are close to the true values and they are

very precise with all true values lying within the standard deviation bounds. Here the nonlinear estimation clearly outperforms the linear one. The linear estimation is biased in τ, δ and all shock standard deviations. Fernádez-Villaverde and Rubio-Ramírez (2004b) report similar point estimates but around 10 times smaller standard deviations. Some differences might of course be due to different simulated data sets.
Nonlinear and linear estimates imply different moments of observables. Fernádez-Villaverde and Rubio-Ramírez (2004a) simulate the model and find this difference to be substantial. This is critical since correlations among observables are often used as an informal model selection criterion.

7.2.4 Marginal Likelihood

Table 11: Marginal Likelihood

	Benchmark Scenario			Risky Scenario		
	@true	1368.2	1365.1		1209.9	-143,651.0
	max	1374.4	1373.7		1213.2	1190.3
p	Δ	Nonlinear	Linear	Δ	Nonlinear	Linear
.1	3.4	-71.5	-74.8	28.8	-111.6	-140.4
.2	3.4	-70.7	-74.1	28.8	-110.9	-139.7
.3	3.5	-70.2	-73.7	28.8	-110.5	-139.3
.4	3.7	-69.6	-73.4	28.8	-110.2	-139.0
.5	3.8	-69.3	-73.1	28.8	-110.0	-138.8
.6	4.0	-68.9	-72.9	29.0	-109.6	-138.6
.7	4.2	-68.5	-72.7	29.0	-109.4	-138.5
.8	4.3	-68.0	-72.3	29.1	-109.3	-138.3
.9	4.8	-67.0	-71.8	29.2	-109.0	-138.2

The econometric variable of interest in this paper is the marginal likelihood given in table 11. It shows the linear versus nonlinear model selection for the benchmark data set and the risky one. In the first two rows are the log likelihoods at the true parameters and the log likelihood maxima of all sequences. In the lower part are the marginal log likelihoods for different quantiles. The nonlinear likelihood at the true values is above zero, higher than the Kalman likelihood and higher at maximum for both data sets. The nonlinear marginal likelihood is higher for the benchmark data set and much higher for the risky set. The nonlinear model is detected to fit both data sets better than the linearized model even if parameters imply a rather linear policy function in the benchmark case.

8 Software

The results were generated using the matrix language Ox version 3.40, see Doornik (2002).[5] It is faster than Matlab and has a C style syntax which simplifies portations to C. The hardware I used is a Pentium 4, 3GHz.
The code consists of three pieces. The first part with 500 lines is a general function approximation tool. It fits Smolyak Chebyshev polynomials to function evaluations on a grid. The second part with 1,200 lines is an integration toolbox which delivers a grid of nodes and associated weights for different densities and approximation levels. The nodes and weights can be uniform, normal, lognormal, t-student, gamma or beta distributed. It delivers Kronecker, nonnested Smolyak nodes and weights as well as nested hierarchical nodes and weights according to Genz and Keister (1996). Any other weight function for which a univariate operator exists or is programmable can be plugged in. Other than normally distributed shocks can be of use for financial market applications or specification tests. The last and main part of the code with 3,500 lines contains the linear and nonlinear solution routines, various filters, the genetic Metropolis-Hastings algorithm, report functions and the marginal likelihood integration. A model file defines first order conditions $\boldsymbol{f}$, state transition functions $\boldsymbol{g}$, measurement functions $\boldsymbol{m}$, the Euler error and various options for the algorithms.
The running times including burn-in sequences for the linear estimation are 10 min and 2 h 20 min for the benchmark and risky parametrization. The nonlinear estimation took 10 and 15 h for $5,000 \times 20 + 50,000$ and $7,000 \times 5 + 50,000$ likelihood evaluations. The tuning sequences for the genetic Metropolis-Hastings parameters took some minutes with the linear Kalman filter.
Smolyak integration can be expected to work efficiently for at least 30 dimensions. Petras (1999) approximated a present value integral with 360 dimensions of different importance in 2 minutes on a 400MHz Pentium machine. Krüger and Kübler (2004) report 14 h for one Smolyak based solution of an OLG model with 30 generations. They used the closed form solution formulas of the Chebyshev coefficients on a Pentium 4, 3GHz. The switch to a matrix based coefficient calculation accelerated in my code the approximation by a factor between 10 and 100.
The interpolation step for the Smolyak Chebyshev polynomials is coded in C. It is the main bottleneck of the whole estimation process and I obtained in C an poor two fold speed up compared to the Ox code for low dimensions and a bit more for higher dimensions. A specialized routine is the first candidate for a further acceleration.
This is especially urgent for the bootstrap filter. The estimation for the simplest model with the nonlinear bootstrap filter takes 20 seconds for each likelihood

[5] Matlab is a trademark of the The MathWorks, Inc. Mathematica is produced by Wolfram-Reasearch, Inc. Maple is a trademark of Waterloo Maple Inc.

evaluation. This sums up for 5 sequences to $(7,000 \times 5 + 50,000) \times 20$ sec ≈ 472 h or 20 days. The Fortran finite element bootstrap filter in Fernádez-Villaverde and Rubio-Ramírez (2004b) takes 88 h for 50,000 draws on a Pentium 4, 3 GHz.
I wrote also the resampling step for the bootstrap filter in C since it needs some nested loops and a vectorization is not possible. The speed gain for each period likelihood contribution is substantial and 40,000 particles are resampled in 0.016 seconds compared to 3.1 in the original Matlab version. The underlying Matlab function `residualR.m` is available on Nando de Freitas' homepage and is a joint work with Arnaud Doucet accompanying the paper van der Merwe, de Freitas, Doucet, and Wan (2001).
I translated the `spgetseq.m` Matlab function by Klimke (2004) for the generation of the Smolyak indices `i` in equation (18). It is part of a Smolyak based linear spline interpolation toolbox. This toolbox cannot extrapolate and is therefore not suited for solving economic models. Extrapolation is critical but arises when the states at the borders of the approximation space transit to next period states outside the border. It sometimes caused the solution algorithm to diverge in the beginning of the maximization draws of the Metropolis-Hastings sequences. This in turn is not critical and after a higher likelihood value is reached divergence does not occur anymore.
I extracted the nodes and weights generation form the fully symmetric Gaussian quadrature code written by Genz and Keister (1996). Their Fortran code is available at Alan Genz's homepage. A Smolyak construction of nonnested univariate quadrature nodes and weights works equally well for nonadaptive integration. This is done with code based on the functions `qnwnorm.m`, ..., `qnwbeta.m` of the *Compecon* Matlab toolbox accompanying Miranda and Fackler (2002) which at the core are translations of quadrature routines in Press, Flannery, Teukolsky, and Vetterling (1988). I also extended the *Compecon* Matlab routines to generate Smolyak based nodes and weights for the paper of Heiss and Winschel (2005).
The linear perturbation solution needs an ordered generalized Schur decomposition. Ox does not provide an ordering option for its Schur decomposition in `decschurgen` and I translated the `qzswitch.m` Matlab function of Christopher Sims.
The marginal likelihood calculation is based on the Matlab function `marginal.m` of Juan F. Rubio-Ramírez. It is part of a Bayesian econometrics course downloadable from his homepage.

9 Conclusion

The framework allows a very general structural analysis of dynamic nonlinear econometric models. Larger models than the one used here might take some days of estimation on a stand-alone computer. Some few lines of code to exchange the current draws of the genetic Metropolis-Hastings algorithm allow the estimation

to be done on a parallel computer cluster. Compared to the approach in Fernádez-Villaverde and Rubio-Ramírez (2004b) my code is around 20 times faster. Since it is written in an interpreted language a further 10 fold speed up in a C or Fortran implementation should be possible. The speed gain of my approach compared to Fernádez-Villaverde and Rubio-Ramírez (2004b) can therefore be expected to be around 200.
The approximation quality of the model solution with Chebyshev polynomials is good and the Smolyak operator substantially reduces computations for approximation and integration already for the smallest possible model.
The nonlinear estimates of the risky parameters are good and better than the benchmark estimates. The estimates of the benchmark parameters are biased and unprecise for the risk parameter τ, autocorrelation ρ and shock variances.
The Gaussian filter is fast and a Gaussian sum approximation of the posterior state densities is a possible cheap road towards the accuracy of the bootstrap filter. Further experience with both filters is desirable for other models.
The genetic extension of the Metropolis-Hastings algorithm is a useful alternative to the random walk algorithm with one sequence and pure random walk shocks. Within the estimation process its handling is rather uncomplicated and the parameters for the algorithm can be tuned fast by a linear estimation. The genetic extension allows an integrated global hill climbing as a first step of a density estimation before sampling around the mode sets in.
The model selection criterion can be calculated fast and it detects the nonlinear nature of the simulated data even if the policy function is almost linear.
The structure of the statistical decision theory and its variable of interest can be fruitfully applied to econometrics for the calculation of specification tests and out-of-sample forecasts.

References

Aruoba, B. S., J. Fernández-Villaverde, and J. F. Rubio-Ramírez (2003): "Comparing Solution Methods for Dynamic Equilibrium Economies," Discussion paper, University of Pennsylvania.

Berger, J., and R. Wolpert (1988): *The Likelihood Principle*. Hayward, California, 2^{nd} edn.

Blanchard, O. J., and C. M. Kahn (1980): "The Solution of Linear Difference Models under Rational Expectations," *Econometrica*, 48, 1305–1312.

Brooks, S. P., and A. Gelman (1998): "General methods for monitoring convergence of iterative simulations.," *Journal of Computational & Graphical Statistics*, 7(4), 434–455.

Chib, S., and E. Greenberg (1995): "Understanding the Metropolis-Hastings Algorithm," *The American Statistician*, 49(4), 327–335.

Doornik, J. A. (2002): *Object-Oriented Matrix Programming Using Ox*. Timberlake Consultants Press and Oxford, London, 3rd edn.

Doucet, A., N. de Freitas, and N. Gordon (eds.) (2001): *Sequential Monte Carlo Methods in Practice*. Springer, New York.

Fernández-Villaverde, J., and J. F. Rubio-Ramírez (2004a): "Estimating Dynamic Equilibrium Economies: Linear versus Nonlinear Likelihood," Discussion paper, University of Pennsylvania.

——— (2004b): "Estimating Nonlinear Dynamic Equilibrium Economies: A Likelihood Approach," Discussion paper, University of Pennsylvania.

Fernández-Villaverde, J., and J. F. Rubio-Ramírez (2004a): "Comparing Dynamic Equilibrium Models to Data: A Bayesian Approach," *Journal of Econometrics*, 123, 153–187.

Friedman, M. (1953): *Essays in Positive Economics*. University of Chicago Press, Chicago and London.

Friedman, M., and L. Savage (1948): "The Utility Analysis of Choices Involving Risk," *Journal of Political Economy*, 56, 279–304.

——— (1952): "The Expected Utility Hypothesis and the Measurability of Utility," *Journal of Political Economy*, 60, 463–474.

Gelfand, A., and D. Dey (1994): "Bayesian Model Choice: Asymptotics and Exact Calculations," *Journal of the Royal Statistical Society, Series B*, 56, 501–514.

Gelman, A., and D. B. Rubin (1992): "Inference from Iterative Simulation Using Multiple Sequences," *Statistical Science*, 7(4), 457–472.

Genz, A. (1986): "Fully Symmetric Interpolatory Rules for Multiple Integrals," *SIAM Journal on Numerical Analysis*, 23, 1273–1283.

Genz, A., and C. Keister (1996): "Fully symmetric interpolatory rules for multiple intergrals over infinite regions with Gaussian weights," *Journal of Computational and Applied Mathematics*, 71, 299–309.

Gerstner, T., and M. Griebel (2003): "Dimension–Adaptive Tensor–Product Quadrature," *Computing*, 71(1), 65–87.

Geweke, J. (1999): "Using Simulation Methods for Bayesian Econometric Models: Inference, Development, and Communication," *Econometric Reviews*, 18, 1–126.

——— (2005): *Contemporary Bayesian Econometrics and Statistics*. John Wiley & Sons, New York.

Gordon, N. J., D. J. Salmond, and A. F. M. Smith (1993): "A novel approach to nonlinear and non-Gaussian Bayesian state estimation," *IEE Proceedings on Radar and Signal Processing*, 140, 107–113.

Hansen, L. P., and J. J. Heckman (1996): "The Empirical Foundations of Calibration," *Journal of Economic Perspectives*, 10(1), 87–104.

Harvey, A. (1985): "Trends and cycles in macroeconomic time series," *Journal of Business and Economic Statistics*, 3, 216–227.

Heiss, F., and V. Winschel (2005): "Smolyak Cubature for Multiple Integration in Estimation Problems," Discussion paper.

Judd, K. (1992): "Projection methods in economics," *Journal of Economic Theory*, 58, 410–452.

Judd, K. L. (1998): *Numerical Methods in Economics*. MIT Press, Cambridge, Mass.

——— (2002): "Perturbation Methods with Nonlinear Changes of Variables," Discussion paper, Hoover Institution, Stanford.

Julier, S. J., and J. K. Uhlmann (1997): "A new extension of the Kalman filter to nonlinear systems," in *The proceedings of AeroSense: The 11th International Symposium on Aerospace/Defence Sensing, Simulation and Controls, Orlando, Florida.*

——— (2002): "The Scaled Unscented Transformation," in *Proceedings of the IEEE American Control Conference*, pp. 4555–4559.

——— (2004): "Unscented Filtering and Nonlinear Estimation," *Proceedings of the IEEE*, pp. 401–422.

Kalman, R. E. (1960): "A New Approach to Linear Filtering and Prediction Problems," *Transactions of the ASME–Journal of Basic Engineering*, 82(Series D), 35–45.

Klein, P. (2000): "Using the generalized Schur form to solve a multivariate linear rational expectations model," *Journal of Economic Dynamics and Control*, 24(10), 1405–1423.

Klimke, A. (2004): "Multilinear sparse grid interpolation in matlab," Discussion paper, University of Stuttgart.

Kotecha, J., and P. Djurić (2003a): "Gaussian Particle Filter," *IEEE Transactions on Signal Processing*, 51, 2592–2601.

——— (2003b): "Gaussian Sum Particle Filter," *IEEE Transactions on Signal Processing*, 51, 2602–2612.

Krüger, D., and F. Kübler (2004): "Computing equilibrium in OLG models with stochastic production," *Journal of Economic Dynamics and Control*, 28, 1411–1436.

Landon-Lane, J. S. (1998): "Bayesian Comparison of Dynamic Macroeconomic Models," Ph.D. thesis, University of Minnesota.

Marimon, R., and A. Scott (eds.) (1999): *Computational Methods for the Study of Dynamic Economies*. Oxford university Press, Oxford.

McGrattan, E. (1998): "Application of weighted residual methods to dynamic economic models," Discussion Paper Staff Report 232, Federal Reserve Bank of Minneapolis.

Miranda, M. J., and P. L. Fackler (2002): *Applied Computational Economics and Finance*. MIT Press, Cambridge, MA.

Morley, J., and J. Piger (2005): "The Importance of Nonlinearity in Reproducing Business Cycle Features," Discussion paper, The Federal Reserve Bank of St.Louis.

Muth, J. F. (1961): "Rational Expectations and the Theory of Price Movements," *Econometrica*, 29, 315–335.

Novak, E., and K. Ritter (1999): "Simple cubature formulas with high polynomial exactness," *Constructive Approximation*, 15, 499–522.

Obstfeld, M., and K. Rogoff (1995): "Exchange rate dynamics redux," *Journal of Political Economy*, 103(3), 624–660.

Patterson, T. (1968): "The Optimum Addition of Points to Quadrature Formulae," *Mathematics of Computation*, 22, 847–856.

Pearl, J. (2000): *Causality*. Cambridge, University Press.

Petras, K. (1999): "Fast Calculation of Coefficients in the Smolyak Algorithm," Discussion paper, Technische Universität Braunschweig.

Press, W. H., B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling (1988): *Numerical Recipes in C: The Art of Scientific Computing*. Cambridge University Press, Cambridge, Massachusetts.

Ruge-Murcia, F. J. (2003): "Methods to Estimate Dynamic Stochastic General Equilibrium Models," Discussion paper, Université de Montréal.

Schmitt-Grohé, S., and M. Uribe (2004): "Solving Dynamic General Equilibrium Models Using a Second-Order Approximation to the Policy Function," *Journal of Economic Dynamics and Control*, (28), 645–658.

Smolyak, S. (1963): "Quadrature and interpolation formulas for tensor products of certain classes of functions," *Soviet Math.Dokl*, 4, 240–243.

Storn, R., and K. Price (1995): "Differential Evolution - a simple and efficient adaptive scheme for global optimization over continuous spaces," Discussion Paper TR-95-012, International Computer Science Institute,, Berkeley.

Tauchen, G., and R. Hussey (1991): "Quadrature-Based Methods for Obtaining Approximate Solutions to Nonlinear Asset Pricing Models," *Econometrica*, 59(2), 371–396.

ter Braak, C. J. (2004): "Genetic algorithms and Markov Chain Monte Carlo: Differential Evolution Markov Chain makes Bayesian computing easy," Discussion paper, Biometris Research Centre, Wageningen University.

van der Merwe, R., N. de Freitas, A. Doucet, and E. Wan (2001): "The Unscented Particel Filter," in *Advances in Neural Information Processing Systems 13*.

Walsh, C. E. (2001): *Monetary Theory and Policy*. MIT Press, Cambridge, Massachusetts.

WASILKOWSKI, G. W., AND H. WOŹNIAKOWSKI (1999): "Weighted tensor product algorithms for linear multivariate problems," *Journal of Complexity*, 15, 402–447.