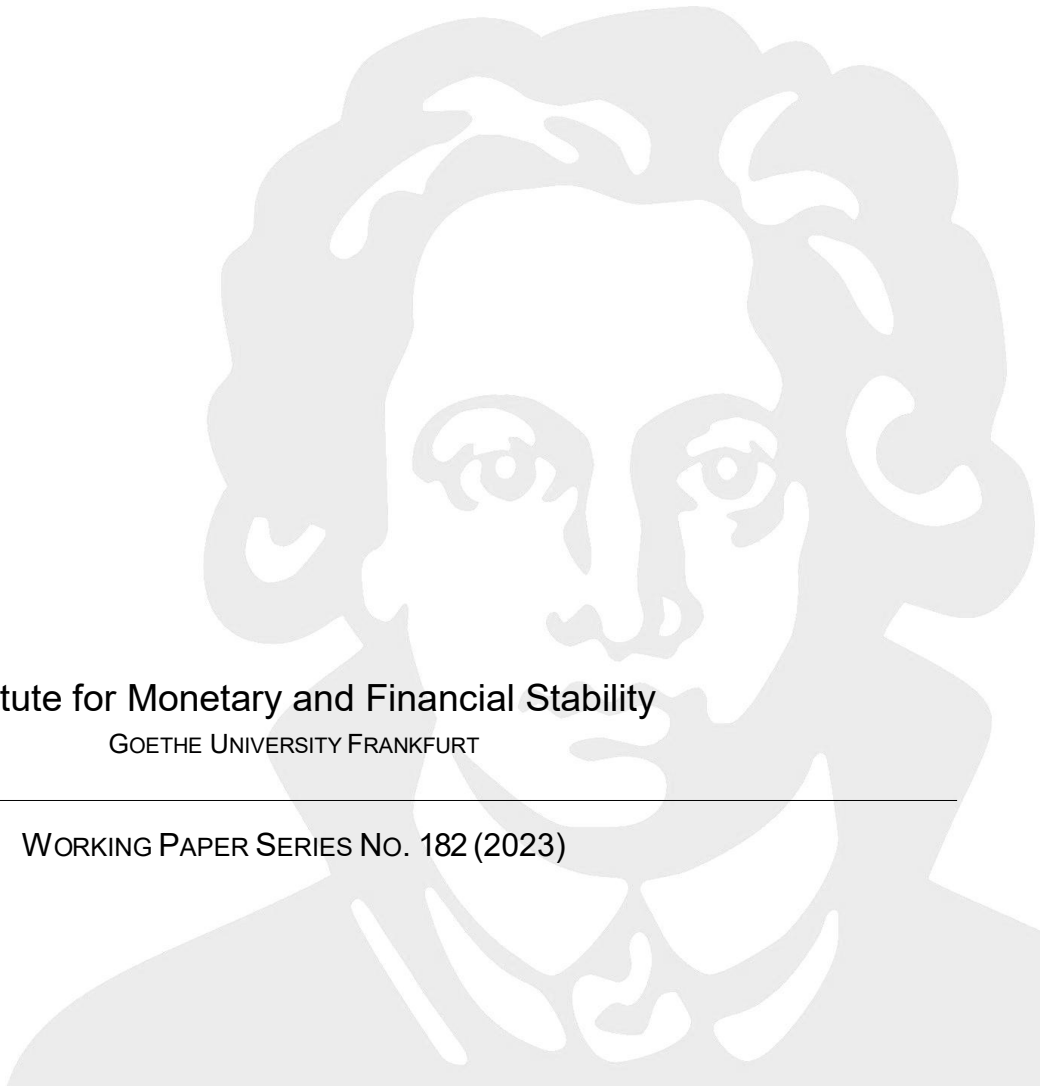
ALEXANDER MEYER-GOHDE

Solving Linear DSGE Models with Bernoulli Iterations



Institute for Monetary and Financial Stability
GOETHE UNIVERSITY FRANKFURT

WORKING PAPER SERIES NO. 182 (2023)

Institute for Monetary and Financial Stability
Goethe University Frankfurt
House of Finance
Theodor-W.-Adorno-Platz 3
D-60629 Frankfurt am Main
www.imfs-frankfurt.de | info@imfs-frankfurt.de

SOLVING LINEAR DSGE MODELS WITH BERNOULLI ITERATIONS

ALEXANDER MEYER-GOHDE

Goethe-Universität Frankfurt and Institute for Monetary and Financial Stability (IMFS)
Theodor-W.-Adorno-Platz 3, 60629 Frankfurt am Main, Germany

ABSTRACT. This paper presents and compares Bernoulli iterative approaches for solving linear DSGE models. The methods are compared using nearly 100 different models from the Macroeconomic Model Data Base (MMB) and different parameterizations of the monetary policy rule in the medium-scale New Keynesian model of Smets and Wouters (2007) iteratively. I find that Bernoulli methods compare favorably in solving DSGE models to the QZ, providing similar accuracy as measured by the forward error of the solution at a comparable computation burden. The method can guarantee convergence to a particular, e.g., unique stable, solution and can be combined with other iterative methods, such as the Newton method, lending themselves especially to refining solutions. *JEL classification codes*: C61, C63, E17

Keywords: Numerical accuracy; DSGE; Solution methods

1. INTRODUCTION

Standard existing methods for solving linear DSGE models predominantly rely on a generalized Schur or QZ decomposition (Moler and Stewart, 1973; Golub and van Loan, 2013) to solve the matrix quadratic equation that underlies the solution. Alternative methods from the applied mathematics literature to solve this matrix quadratic equation have yet to be systematically studied in a DSGE context. This paper fills part of that gap, collecting Bernoulli-based solution methods, combining them with Newton based methods (see Meyer-Gohde and Saecker (2022)) for matrix quadratic problems and applying them to the solution of linear DSGE models. The iterative nature and ability of Bernoulli

E-mail address: meyer-gohde@econ.uni-frankfurt.de.

Date: May 3, 2023.

I am grateful to Johanna Saecker for useful comments and suggestions and to Elena Schlipphack and Maximilian Thomin for invaluable research assistance. Any and all errors are entirely my own. This research was supported by the DFG through grant nr. 465469938 "Numerical diagnostics and improvements for the solution of linear dynamic macroeconomic models".

methods to be formulated such that they converge to the minimal solvent (solution with the smallest eigenvalues in magnitude) allow them to perform favorably compared with QZ-based methods - particularly when combined with Newton methods and their asymptotic quadratic convergence. Precisely this iterative characteristic also enables the methods I introduce to linear DSGE models to correct insufficiently accurate solutions of economic consequence as presented in Meyer-Gohde (2022).

Bernoulli-based functional iterations, familiar to economists in root-finding settings - see, e.g., Judd (1998), are an alternative to QZ-based methods but have not yet been examined for solving linear DSGE models.[1] Bernoulli iterations following Dennis, Jr., Traub, and Weber (1978), have been explored and expanded by Higham and Kim (2000) and Bai and Gao (2007), which have the advantage that the iterations can be formulated to recover a particular solution, or solvent, for example the unique stable solution often sought in a DSGE context. Meyer-Gohde and Saecker (2022) examine a number of Newton based methods and I also derive and examine iterations that combine both Bernoulli and Newton methods. Similarly, I take the insights of Higham and Kim (2001), who incorporate exact line searches into a Newton algorithm and show it improves global convergence by making it faster and more reliable, to improve on the slow convergence (i.e., many iterations) of standard Bernoulli algorithms. Furthermore, they derive a conditioning number and bound the backward error, as reviewed and applied to a DSGE context in Meyer-Gohde (2022) - whose practical forward error bounds are used to evaluate alternate solutions here.

In this paper, I present eleven different Bernoulli-based solution algorithms using a unified notation and for the application to solving linear DSGE models as an alternative to QZ-based methods. I engage in a number of experiments to compare the algorithms to QZ-based methods[2] and Meyer-Gohde and Saecker's (2022) Newton algorithms, following exactly their experiments to ensure comparability. First I apply the different methods to the models in the Macroeconomic Model Data Base (MMB) (see Wieland, Cwik, Müller, Schmidt, and Wolters, 2012; Wieland, Afanasyeva, Kuete, and Yoo, 2016), comparing the performance to the QZ-based method of Dynare and the baseline Newton method both unconditionally (i.e., replacing the QZ method) and then as a refinement (i.e., initializing

[1] In fact, the only functional iteration I am aware of is Binder and Pesaran's (1997) "fully recursive algorithm" that can be reformulated as a functional iteration on the matrix quadratic.

[2] I use Dynare's (Adjemian, Bastani, Juillard, Mihoubi, Perendia, Ratto, and Villemot, 2011) implementation of the QZ method, documented in Villemot (2011), for comparison.

the iterative methods with the solution generated from QZ). I find that the baseline Bernoulli method always converges to the unique stable solution, but provides a solution of the same order of magnitude of accuracy at an order of magnitude higher computational cost. The different extended methods generally perform favorably - the exception being Bai and Gao's (2007) modified Bernoulli iteration that is several orders of magnitude more slow and frequently faces convergence issues - trading the guaranteed convergence of the baseline algorithm off against more accuracy and/or less computational costs.

The algorithms are iterative in nature, enabling them to refine the solutions provided by the QZ method. Using the QZ solution from Dynare to initialize, the methods improve the accuracy of the solution at a modest addition computational cost, with convergence of all of the algorithms to the unique stable solvent for all of the models in the MMB, except Bai and Gao's (2007) modified Bernoulli and columnwise Newton-Bernoulli combinations. This iterative nature also lends itself to iterative parameter experiments or estimations and I compare the algorithms with the QZ method and Meyer-Gohde and Saecker's (2022) Newton algorithm in solving for different parameterizations of the monetary policy rule in the celebrated Smets and Wouters (2007) model of the US economy. Filling in a grid with different values of the reaction of the nominal interest rate rule to inflation and real activity, whereas the QZ method starts anew at each parameterization, iterative methods can use the solution from the previous, nearby parameterization to initialize the algorithm. As the density of the grid increases, all of the methods eventually surpass QZ by roughly an order of magnitude in terms of computation cost.

The paper is organized as follows: section 2 lays out the general DSGE model and the unknown solution. In section 3, I present the set of different Bernoulli methods from the applied mathematics literature in a unified notation as they apply to the class of DSGE models. Section 4 examines practical and theoretical considerations such as the choice of initial value, accuracy, and convergence. In section 5, I compare the different Bernoulli methods to the standard QZ method and Newton method of Meyer-Gohde and Saecker (2022) in two applications, one using the MMB of 99 different models and the second over a range of parameterizations within the Smets and Wouters (2007) model. Finally, section 6 concludes.

2. Problem Statement

Standard numerical solution packages available to economists and policy makers—e.g., Dynare (Adjemian, Bastani, Juillard, Mihoubi, Perendia, Ratto, and Villemot, 2011),

Gensys (Sims, 2001), (Perturbation) AIM (Anderson and Moore, 1985; Anderson, Levin, and Swanson, 2006), Uhlig's Toolkit (Uhlig, 1999) and Solab (Klein, 2000)—all analyze models that in some way or another can be expressed in the form of the nonlinear functional equation

$$0 = E_t[f(y_{t+1}, y_t, y_{t-1}, \varepsilon_t)] \tag{1}$$

The model equations (optimality conditions, resource constraints, market clearing conditions, etc.) are represented by the n_y-dimensional vector-valued function $f : \mathbb{R}^{n_y} \times \mathbb{R}^{n_y} \times \mathbb{R}^{n_y} \times \mathbb{R}^{n_e} \to \mathbb{R}^{n_y}$; $y_t \in \mathbb{R}^{n_y}$ is the vector of n_y endogenous variables; and $\varepsilon_t \in \mathbb{R}^{n_e}$ the vector of n_e exogenous shocks with a known distribution, where n_y and n_e are positive integers ($n_y, n_e \in \mathbb{N}$).

The solution to (1) is sought as the unknown function

$$y_t = y(y_{t-1}, \varepsilon_t), \quad y : \mathbb{R}^{n_y + n_e} \to \mathbb{R}^{n_y} \tag{2}$$

a function in the time domain that maps states, y_{t-1} and ε_t, into endogenous variables, y_t. An analytic form for (2) is rarely available and researchers and practitioners are compelled to find approximative solutions. However, a steady state, $\overline{y} \in \mathbb{R}^{n_y}$ a vector such $\overline{y} = y(\overline{y}, 0)$ and $0 = f(\overline{y}, \overline{y}, \overline{y}, 0)$ can frequently be recovered, either analytically or numerically, providing a point of expansion around which local solutions may be recovered.

A first-order, or linear, approximation of (1) at the steady state delivers,

$$0 = AE_t[y_{t+1}] + By_t + Cy_{t-1} + D\varepsilon_t \tag{3}$$

where A, B, C, and D are the derivatives of f in (1) with respect to its arguments and, recycling notation, the y's in (3) refer to (log) deviations of the endogenous variables from their steady states, $\overline{y}$.

In analogy to (2), the standard approach to finding a solution to the linearized model (3) is to find a linear solution in the form

$$y_t = P\, y_{t-1} + Q\, \varepsilon_t \tag{4}$$

a recursive solution in the time domain–solutions that posit y_t as a function of its own past, y_{t-1}, and exogenous innovations, ε_t.

Inserting (4) into (3) and taking expectations ($E_t[\varepsilon_{t+1}] = 0$), yields the restrictions

$$0 = AP^2 + BP + C, \quad 0 = (AP + B)Q + D \tag{5}$$

Generally, a unique P with eigenvalues inside the closed unit circle is sought. Lan and Meyer-Gohde (2014) prove the latter can be uniquely solved for Q if such a P can be found. Hence, the hurdle is the former, matrix quadratic equation.

Most linear DSGE methods use a generalized Schur or QZ decomposition (Moler and Stewart, 1973; Golub and van Loan, 2013) of the companion linearization of (3)[3] in some form or another. I will take a different route and instead now solve for P in (5) using Bernoulli iterations.

3. Bernoulli Iterations for Linear DSGE models

I will begin by analyzing a univariate equation, see, e.g., Higham (2002, p. 508) for the functional iteration to find the unstable solution and Judd (1992, pp. 152-153) for functional iterations in general fix point problems for economists, to fix ideas and illustrate how Bernoulli functional iterations can be used to solve quadratic equations.

The problem generated by (5) is a (matrix) quadratic problem. To fix ideas, consider its univariate equivalent

$$0 = ax^2 + bx + c \tag{6}$$

where I consider (in accordance with the DSGE model), a, b, and $c \in \mathbb{R}^1$. A functional iteration will reformulate this as

$$x = f(x) \tag{7}$$

giving an iterative procedure to generate a solution

$$x_{j+1} = f\left(x_j\right), \text{ with some } x_0 \tag{8}$$

From the quadratic equation above, there are a number of possibilities,

$$f(x) = -\frac{c}{ax+b} \tag{9}$$

$$f(x) = -\frac{b+\frac{c}{x}}{a} \tag{10}$$

$$f(x) = -\frac{ax^2+c}{b} \tag{11}$$

[3]For a presentation of the QZ decomposition for solving linear DSGE models with the method of undetermined coefficients and a multivariate pivoted Blanchard (1979) approach, see Meyer-Gohde (2022).

and so forth. I will focus on the first $f(x) = -\frac{c}{ax+b}$ as it is the univariate counterpart of what follows. Hence

$$x_{k+1} = f(x_k) = -\frac{c}{ax_k + b}, \; k \in \mathbb{N}^0, x_0 \text{ given} \tag{12}$$

Characterize the two solutions via

$$(s_1 x - t_1)(s_2 x - t_2) = \underbrace{s_1 s_2}_{a} x^2 \underbrace{-(s_1 t_2 + s_2 t_1)}_{+b} x + \underbrace{t_1 t_2}_{c} \tag{13}$$

where the solutions are

$$x_i = \left\{ \frac{t_i}{s_i}, \text{ if } s_i \neq 0; \infty, \text{ if } s_i = 0; \emptyset, \text{ if } s_i = t_i = 0; i = 1,2 \right\} \tag{14}$$

Following, e.g., Judd (1998, pp. 165-166), local convergence requires $\left|f'(x_k)\right| < 1$ and as

$$f'(x_k) = \frac{ac}{(ax_k + b)^2} = \frac{s_1 s_2 t_1 t_2}{(s_1 s_2 x_k - (s_1 t_2 + s_2 t_1))^2} \tag{15}$$

for, say, $x_1 = \frac{t_1}{s_1}$

$$f'(x_1) = \frac{s_1 s_2 t_1 t_2}{\left(s_1 s_2 \frac{t_1}{s_1} - (s_1 t_2 + s_2 t_1)\right)^2} = \frac{s_1 s_2 t_1 t_2}{(s_1 t_2)^2} = \frac{s_2 t_1}{s_1 t_2} = \frac{x_1}{x_2} \tag{16}$$

and hence the iteration is convergent if

$$\left|f'(x_1)\right| = \left|\frac{x_1}{x_2}\right| < 1 \tag{17}$$

or there is local convergence to the minimal (smaller in modulus) root for $f(x) = -\frac{c}{ax+b}$.

This highlights a significant advantage over Newton-based methods, namely that the functional iteration can be tailored to deliver convergence of the algorithm to a particular root: for example, $f(x) = -\frac{c}{ax+b}$ to the minimal (as above) and $f(x) = -\frac{b+\frac{c}{x}}{a}$ to the dominant (Higham and Kim, 2000) solution. As a particular solution or a solution with particular properties (namely the minimal solution in a saddle point stable problem) is sought in DSGE models, the Bernoulli iteration $x_{k+1} = f(x_k) = -\frac{c}{ax_k+b}$ is potentially very useful in solving DSGE models.

Turning now to the matrix problem, I will formalize the matrix quadratic equation in (5). For A, B, and $C \in \mathbb{R}^{n_y \times n_y}$, a matrix quadratic $M(P) : \mathbb{C}^{n_y \times n_y} \to \mathbb{C}^{n_y \times n_y}$ is defined as

$$M(P) \equiv AP^2 + BP + C \tag{18}$$

with its solutions, called solvents, given by $P \in \mathbb{C}^{n_y \times n_y}$ if and only if $M(P) = 0$. The eigenvalues of the solvent, called latent roots of the associated lambda matrix[4] $M(\lambda) : \mathbb{C} \to$

[4]See, e.g., Dennis, Jr., Traub, and Weber (1976, p. 835) or Gantmacher (1959, vol. I, p. 228).

$\mathbb{C}^{n\times n}$ (here of degree two), are given via

$$M(\lambda) \equiv A\,\lambda^2 + B\,\lambda + C \tag{19}$$

The latent roots are (i) values of $\lambda \in \mathbb{C}$ such that $\det M(\lambda) = 0$ and (ii) $n_y - \text{rank}(A)$ infinite roots. An explicit link between the quadratic matrix problem and the quadratic eigenvalue problem is given via

$$\lambda \in \mathbb{C} : \left(A\lambda^2 + B\lambda + C\right)x = 0 \text{ for some } x \neq 0 \tag{20}$$

which has been reviewed extensively by Tisseur and Meerbergen (2001) and for which Hammarling, Munro, and Tisseur (2013) provide a comprehensive method to improve the accuracy of its solutions.

The matrix quadratic (18) can be expanded following Higham and Kim (2001) as

$$\begin{aligned} M(P+\Delta P) &= A\,(P+\Delta P)^2 + B\,(P+\Delta P) + C && (21)\\ &= A\,P^2 + B\,P + C + A\,\Delta P^2 + A\,(P\Delta P + \Delta P P) + B\,\Delta P && (22)\\ &= M(P) + (A\,\Delta P P + (A\,P + B)\,\Delta P) + A\,\Delta P^2 && (23)\\ &= M(P) + \mathscr{D}_P(\Delta P) + A\,\Delta P^2 && (24) \end{aligned}$$

where $\mathscr{D}_P M(\Delta P)$ is the Fréchet derivative of M at P in the direction ΔP.

3.1. Baseline Bernoulli Iteration

The multivariate counterpart to the algorithm $x_{k+1} = f(x_k) = -\frac{c}{ax_k+b}$ above is the following baseline Bernoulli method for the minimal solvent. Beginning with (18)

$$0 = (A\,P + B)P + C \tag{25}$$

and defining the iteration via

$$0 = \left(A\,P_j + B\right)P_{j+1} + C \tag{26}$$

gives the baseline Bernoulli method

$$P_{j+1} = -\left(A\,P_j + B\right)^{-1} C \tag{27}$$

summarized in the following algorithm

Baseline Bernoulli Method

- Given A, B, C, an initial P_0, and a convergence criterion ϵ
- While criterion$(P_j) > \epsilon$
  (1) Set $P_{j+1} = -\left(AP_j + B\right)^{-1} C$
  (2) Advance $j = j + 1$
- Return P_j

Higham and Kim (2000) show that the above recursion converges asymptotically at a linear rate to the minimal solvent, but only under the assumption that this solvent is invertible (e.g., ruling out zero latent roots in (19)) and that a dominant solvent also exists (e.g., ruling out "infinite" latent roots in (19)), both of which abound in the DSGE literature. I return to this issue in section 4 and provide a proof that this recursion converges at least locally to the minimal solvent when this solvent is the unique stable solution.

3.2. Bai and Gao's (2007) Modified Bernoulli Method

Bai and Gao (2007) note that the Bernoulli recursion

$$P_{j+1} = -\left(AP_j + B\right)^{-1} C \tag{28}$$

solves n linear equations at each iteration

$$\left(AP_j + B\right)P_{j+1} = -C \Leftrightarrow \left(AP_j + B\right)P_{i,j+1} = -C_i \text{ for } i = 1,...,n \tag{29}$$

and that when solving these individual equations in sequence, one can use the "real-time" value of P, updated to those columns that precede the current column

$$\left(A\tilde{P}_{i,j} + B\right)P_{i,j+1} = -C_i \text{ for } i = 1,...,n \tag{30}$$

where

$$\tilde{P}_{i,j} = \begin{bmatrix} P_{1,j} & \dots & P_{i,j} & P_{i+i,j+1} & \dots & P_{n,j+1} \end{bmatrix} \tag{31}$$

Instead of solving n systems at each iteration

$$P_{i,j+1} = -\left(A\tilde{P}_{i,j} + B\right)^{-1} C_i \text{ for } i = 1,...,n \tag{32}$$

they use the Sherman-Morrison formula for updating matrices by noticing that $\tilde{P}_{i,j}$ is a rank one correction of $\tilde{P}_{i-1,j}$. Moving through the columns successively gives

$$P_{i+1,j+1} = -\left(AP_j + B\right)^{-1} C_{i+1} + \sum_{l=1}^{i} \frac{e_l' q_{l,j}}{1 + e_l' p_{l,j}} p_{l,j} \tag{33}$$

where

$$u_{i,j} = A\left(P_{i,j+1} - P_{i,j}\right) \tag{34}$$

$$q_{l,j} = \left(AP_j + B\right)^{-1} u_{i,j} \tag{35}$$

$$p_{l,j} = \left(AP_j + B\right)^{-1} C_{i+1} \tag{36}$$

and e_i' is a vector of zeros with a one in the i'th column. This gives the following modified Bernoulli algorithm

Modified Bernoulli Method

- Given A, B, C, an initial P_0, and a convergence criterion ϵ
- While criterion$(P_j) > \epsilon$
  (1) For $i = 1:n$ solve for $P_{i,j+1}$ via
  $$P_{i,j+1} = -\left(AP_j + B\right)^{-1} C_i \tag{37}$$
  (2) For $i = 1:n-1$ update $P_{i,j+1}$ recursively via
    (a) $u_{i,j} = A\left(P_{i,j+1} - P_{i,j}\right)$
    (b) $p_{l,j} = \left(AP_j + B\right)^{-1} C_{i+1}$
    (c) $q_{l,j} = P_{i+1,j+1}$
    (d) for $l = i+1:n$
    $$P_{l,j+1} = P_{l,j+1} + \frac{e_i' q_{i,j}}{1 + e_i' p_{i,j}} p_{i,j} \tag{38}$$
  (3) Advance $j = j+1$
- Return P_j

Bai and Gao (2007) provide proof that this recursion converges to the minimal solvent, again under the assumption that a minimal and dominant solvent both exist. While they argue that this recursion outperforms the baseline Bernoulli method, their own examples show that this outperformance is minimal.

3.3. Bernoulli Line Search

Newton based methods, see Meyer-Gohde and Saecker (2022), allow for the straightforward incorporation of line search routines to find optimal increments, see Higham and Kim (2001). A line search to scale the increments can protect against perniciously large steps and accelerate convergence when the increments are timidly small. Bernoulli iterations generally require many more iterations than Newton based methods and their rate of convergence is governed by the ratio of the largest in modulus stable and smallest in modulus unstable latent roots, e.g. Higham and Kim (2000) - hence the primary difficulty will be slow convergence associated with timidly small increments.

The Bernoulli methods above, however, are functional iterations that work directly on the solution P. But simple reformulation reveals the implied increment in a Bernoulli step

$$P_{j+1} = -(AP_j + B)^{-1}C \tag{39}$$

$$\equiv \Delta P_j + P_i \tag{40}$$

$$= \underbrace{-(AP_j + B)^{-1}}_{K(P_j)} M(P_j) + P_j \tag{41}$$

In a Newton context, Higham and Kim (2001) motivate an exact line search by the inaccuracies of the linear approximation that delivers the Newton step: if P_j is far from a solvent ($P : M(P) = 0$), the update $P_{j+1} = P_j + \Delta P_j$ might be farther from a solvent than P_j. The same logic holds with the Bernoulli iteration, albeit for a different reason: with the linear convergence of the algorithm, the update $P_{j+1} = P_j + \Delta P_j$ is likely to be too conservative an update. I propose a line search, a multiple of the Bernoulli step above, $P_{j+1} = t\Delta P_j + P_j$ where t is an appropriate scalar. Obviously, if $t = 1$, the baseline Bernoulli iteration is recovered. Following Higham and Kim (2001) and Meyer-Gohde and Saecker (2022), I select the multiple of the Bernoulli step by finding a t that minimizes the merit function

$$t = \underset{x \geq 1}{\operatorname{argmin}} \| M(P + x\Delta P) \|_F^2 \tag{42}$$

The term $M(P+x\Delta P)$ can be expressed explicitly as

$$M(P+x\Delta P) = A(P+x\Delta P)^2 + B(P+x\Delta P) + C \tag{43}$$

$$= AP^2 + BP + C + Ax^2\Delta P2 + A(Px\Delta P + x\Delta P \cdot P) + Bx\Delta P \tag{44}$$

$$= M(P) + x^2 A\Delta P^2 + x\underbrace{(A(P\Delta P + \Delta P \cdot P) + B\Delta P)}_{\mathscr{D}_P(\Delta P)} \tag{45}$$

where $\mathscr{D}_P(\Delta P)$ is the Fréchet derivative of M at P in the direction ΔP. Higham and Kim (2001) show that this particular choice of merit function is convenient for Newton based algorithms as the Newton step sets the Fréchet derivative to the negative of the matrix quadratic ($\mathscr{D}_P(\Delta P) + M(P) = 0$) by construction allowing $M(P+x\Delta P)$ to be written as $M(P+x\Delta P) = (1-x)M(P) + x^2 A\Delta P^2$. With the Bernoulli iteration, we are not quite as fortunate, as

$$D_p(\Delta P) = (AP+B)\Delta P + A\Delta P \cdot P \tag{46}$$

$$= -K(P)^{-1}\Delta P + A\Delta P \cdot P \tag{47}$$

$$= -M(P) + A\Delta P \cdot P \tag{48}$$

and hence

$$M(P+x\Delta P) = M(P) + x^2 A\Delta P^2 - xM(P) + xA\Delta P \cdot P \tag{49}$$

$$= (1-x)M(P) + xA\Delta P(x\Delta P + P) \tag{50}$$

The merit function (42) and its first derivative are thus, see the appendix for details,

$$g(x) \equiv \|M(P+x\Delta P)\|_F^2 = \gamma x^4 + (\xi - \sigma)x^3 + \left(\alpha + \beta - \delta + \sigma\right)x^2 + (\delta - 2\alpha)x + \alpha \tag{51}$$

$$g'(x) = 4\gamma x^3 + 3(\xi - \sigma)x^2 + 2\left(\alpha + \beta - \delta + \sigma\right)x + \delta - 2\alpha \tag{52}$$

where $\alpha = \|M(P)\|_F^2$, $\beta = \|A\Delta P \cdot P\|_F^2$, $\gamma = \left\|A(\Delta P)^2\right\|_F^2$, $\xi = \operatorname{tr}\left((A\Delta P \cdot P)^* A(\Delta P)^2 + \left(A(\Delta P)^2\right)^* A\Delta P \cdot P\right)$, $\sigma = \operatorname{tr}\left(M(P)^* A(\Delta P)^2 + \left(A(\Delta P)^2\right)^* M(P)\right)$, and $\delta = \operatorname{tr}\left(M(P)^* A\Delta P \cdot P + (A\Delta P \cdot P)^* M(P)\right)$. As $g(x)$ is a quartic polynomial it has at most two minima.

As the goal is to minimize $g(x)$ over the closed one to infinity range, I need to find minima and compare their values to $g(1)$. Finding extrema corresponds to finding zeros of the cubic polynomial $g'(x)$ in this range. Implementing t from (42) is straightforward as either there is a single real zero of $g'(x)$ which lies in the range and is a minimum of $g(x)$ (as $\alpha > 0$) or $g'(x)$ has three real zeros, of which at most two correspond to minima of

$g(x)$. Hence, finding the zeros of $g'(x)$ and comparing the associated values of $g(x)$ with the value of $g(1)$ enables t from (42) to be readily found.

This gives the Bernoulli procedure with exact line searches as

Exact Line Searches

- Given A, B, C, an initial P_0, and a convergence criterion ϵ
- While criterion$(P_j) > \epsilon$
  (1) Solve for $\Delta P_j = -(AP_j + B)^{-1} M(P_j)$
  (2) Solve for t_j in

$$t_j = \underset{x \geq 1}{\operatorname{argmin}} \left\| M(P_j + x\Delta P_j) \right\|_F^2 \tag{53}$$

  (3) Set $P_{j+1} = P_j + t_j \Delta P_j$
  (4) Advance $j = j + 1$
- Return P_j

Line searches have been shown by Higham and Kim (2001) to not interfere with the asymptotic convergence rate, hence the expectation of this algorithm is a factor but not order increase in the convergence speed.

3.4. Combining Bernoulli and Newton

From the previous algorithm, the Bernoulli iteration $P_i \to P_{j+1}$ can be described via an increment, $\Delta_B P_j$,

$$P_{B,j+1} = \Delta_B P_j + P_i \tag{54}$$

The same holds for the Newton methods studied in Meyer-Gohde and Saecker (2022) with the increment $\Delta_N P_j$,

$$P_{N,j+1} = \Delta_N P_j + P_i \tag{55}$$

where $\Delta_N P_j$ is the increment that sets the first-order in ΔP approximation of the Fréchet derivative of M at P in the direction ΔP, see (24), to zero.

This observation motivates the following iteration

$$P_{j+1} = s_j \Delta_B P_j + \left(1 - s_j\right) \Delta_N P_j + P_j, \ 0 \leq s_j \leq 1 \tag{56}$$

with an average of the Bernoulli and Newton increments being weighted by s_j. A simple choice of weights stems from the observations that motivated line searches both for Newton and Bernoulli iterations. For Newton based methods, there is a potential of

taking too large steps and in the wrong (i.e., not towards the minimal solvent) and for Bernoulli too small steps. Hence, one would like to take the potentially larger Newton step when it is going in the same general direction that the Bernoulli step would take and to take the Bernoulli step when the Newton step would go in a different direction (presumably to another solvent).

Hence, measuring the angle, θ between the two increments via (Horn and Johnson, 2012, p.15)

$$\cos\theta = \frac{\text{vec}(\Delta_N P_j)' \text{vec}(\Delta_B P_j)}{\|\Delta_N P_j\|_2 \|\Delta_B P_j\|_2} \tag{57}$$

the weight between the increments is defined as

$$s_j \equiv \arccos(\theta)/\pi \tag{58}$$

So when the angle between the two increments is zero (i.e., they are moving in the same direction), the cosine of θ is one, θ is zero, and s_j is zero: $P_{j+1} = \Delta_N P_j + P_i$, the Newton step with its asymptotically quadratic rate of convergence is taken. When this angle is π (i.e., the increments are moving in opposite directions), the cosine of θ is -1, θ is π, and s_j is one: $P_{j+1} = \Delta_B P_j + P_i$, the Bernoulli step with its asymptotic convergence to the minimal solvent (see the next section for a proof) is taken.

Alternatively, the increments could be weighted columnwise, that is comparing the increments associated with the solutions for the individual variables in y_t,

$$P_{j+1} \equiv \begin{bmatrix} P_{1,j+1} & \dots & \dots & P_{n,j+1} \end{bmatrix} \tag{59}$$

$$= \begin{bmatrix} s_{1,j}\Delta_B P_{1,j} + (1 - s_{1,j})\Delta_N P_{1,j} & \dots & \dots & s_{n,j}\Delta_B P_{n,j} + (1 - s_{n,j})\Delta_N P_{n,j} \end{bmatrix} + P_j \tag{60}$$

where the weight between the columns of the increments ($0 \leq s_{i,j} \leq 1$) is defined as

$$s_{i,j} \equiv \arccos(\theta_i)/\pi \tag{61}$$

via the angle, θ_i between the i'th columns of the two increments via (Horn and Johnson, 2012, p.15)

$$\cos\theta_i = \frac{\Delta_N P'_{i,j} \Delta_B P_{i,j})}{\|\Delta_N P_{i,j}\|_2 \|\Delta_B P_{i,j}\|_2} \tag{62}$$

or the weighting above could be tilted towards one or the other increment

$$\tilde{s}_j(p) = s_j^p \tag{63}$$

with an exponent $p < 1$ shifting the weight in favor of the Bernoulli increment (and hence $\tilde{s}_j(p)$ replacing s_j in (56)), perhaps to reduce the likelihood of the algorithm converging to a solvent other than the unique stable one.

This gives a combined Bernoulli-Newton procedure as

Combined Bernoulli-Newton

- Given A, B, C, an initial P_0, a convergence criterion ϵ, and a weight tilt p
- While criterion$(P_j) > \epsilon$
  (1) Solve for $\Delta_B P_j = -(AP_j + B)^{-1} M(P_j)$
  (2) Solve for $\Delta_N P_j$ in

$$A\,\Delta_N P_j P_j + \left(A\,P_j + B\right)\Delta_N P_j = -M(P_j) \tag{64}$$

  (3) Determine $\cos\theta$ (or $\cos\theta_i$ for all columns i)
  (4) Calculate $s_j = \arccos(\theta)/\pi$ (or $s_{i,j} = \arccos(\theta_i)/\pi$ for all columns i)
  (5) Set $P_{j+1} = s_j^p \Delta_B P_j + \left(1 - s_j^p\right)\Delta_N P_j + P_j$
  (or $P_{i,j+1} = s_{i,j}^p \Delta_B P_{i,j} + \left(1 - s_{i,j}^p\right)\Delta_N P_{i,j} + P_{i,j}$ for all columns i)
  (6) Advance $j = j + 1$
- Return P_j

While this procedure will hopefully yield the best of both worlds: convergence to the desired solvent via Bernoulli and quadratic convergence via Newton, it might also do just the opposite - combining the linear convergence of Bernoulli to the unpredictable solvent of Newton. Tilting the weight towards one or the other procedure gives the user flexibility, but it is still not clear a priori how that tilt might be chosen to deliver a procedure with the desired properties.

3.5. Bernoulli and Newton Line Search

As the weighting of the Bernoulli and Newton increments relied on the same intuition that guided their respective line search algorithms, the next logical step would be to combine both the line searches for the optimal magnitudes of the respective increments and then weight the two to produce a combined Bernoulli and Newton increment with line searches. That is, if combining Bernoulli and Newton might bring the slow convergence of Bernoulli together with the unpredictable convergence destination of Newton, performing line searches on the steps of both of these might attenuate this danger. From above, the step size of Bernoulli can be increased past 1 to improve the speed of convergence, and

from Meyer-Gohde and Saecker (2022) the step size of Newton can be adjusted optimally to balance the danger of taking too large of steps.

This gives a combined Bernoulli-Newton procedure with line searches as

Combined Bernoulli-Newton with Line Searches

- Given A, B, C, an initial P_0, a convergence criterion ϵ, and a weight tilt p
- While criterion$(P_j) > \epsilon$
  (1) Solve for $\Delta_B P_j = -(AP_j + B)^{-1} M(P_j)$
  (2) Solve for $t_{B,j}$ in

$$t_{B,j} = \underset{x \geq 1}{\operatorname{argmin}} \left\| M(P_j + x\Delta_B P_j) \right\|_F^2 \tag{65}$$

  (3) Solve for $\Delta_N P_j$ in

$$A\,\Delta_N P_j P_j + \left(A\,P_j + B\right)\Delta_N P_j = -M(P_j) \tag{66}$$

  (4) Solve for $t_{N,j}$ in

$$t_{N,j} = \underset{x \in [0,2]}{\operatorname{argmin}} \left\| M(P_j + x\Delta_N P_j) \right\|_F^2 \tag{67}$$

  (5) Determine $\cos\theta$
  (6) Calculate $s_j = \arccos(\theta)/\pi$
  (7) Set $P_{j+1} = s_j^p t_{B,j} \Delta_B P_j + \left(1 - s_j^p\right) t_{N,j} \Delta_N P_j + P_j$
  (8) Advance $j = j + 1$
- Return P_j

As above, the underlying steps in the procedure would tend to limit the drawbacks of the two procedures on their own. Yet, it is not a priori certain how they will perform when combined.

3.6. Optimal Bernoulli and Newton

The final set of algorithms explicitly take the optimality approach when determining s_j, minimizing the same merit function used in the line searches above. First for the initial increments

$$s = \underset{0 \leq x \leq 1}{\operatorname{argmin}} \left\| M(x\Delta_B P + (1-x)\Delta_N P + P) \right\|_F^2 \tag{68}$$

and then for the line-search optimized increments

$$s = \underset{0 \leq x \leq 1}{\operatorname{argmin}} \left\| M(x t_B \Delta_B P + (1-x) t_N \Delta_N P + P) \right\|_F^2 \tag{69}$$

where

$$t_i = \underset{x \in X_i}{\operatorname{argmin}} \|M(x\Delta_i P + P)\|_F^2, \text{ for } i = B, N \tag{70}$$

where X_B restricts x to be greater or equal to one, see above, and X_N restricts x to be between zero and two (see Meyer-Gohde and Saecker, 2022).

This gives an optimally (in the sense of the merit function) combined Bernoulli-Newton procedure

Optimal Bernoulli-Newton

- Given A, B, C, an initial P_0, and a convergence criterion ϵ
- While criterion$(P_j) > \epsilon$
  - (1) Solve for $\Delta_B P_j = -(AP_j + B)^{-1} M(P_j)$
  - (2) Solve for $\Delta_N P_j$ in

$$A\,\Delta_N P_j P_j + \left(A\,P_j + B\right)\Delta_N P_j = -M(P_j) \tag{71}$$

  - (3) Either set $t_{B,j} = t_{N,j} = 1$, or
    - (a) Solve for $t_{B,j}$ in

$$t_{B,j} = \underset{x \geq 1}{\operatorname{argmin}} \left\|M(P_j + x\Delta_B P_j)\right\|_F^2 \tag{72}$$

    - (b) Solve for $t_{N,j}$ in

$$t_{N,j} = \underset{x \in [0,2]}{\operatorname{argmin}} \left\|M(P_j + x\Delta_N P_j)\right\|_F^2 \tag{73}$$

  - (4) Solve for s_j

$$s_j = \underset{0 \leq x \leq 1}{\operatorname{argmin}} \left\|M(xt_{B,j}\Delta_B P_j + (1-x)t_{N,j}\Delta_N P_j + P_j)\right\|_F^2 \tag{74}$$

  - (5) Set $P_{j+1} = s_j t_{B,j}\Delta_B P_j + \left(1 - s_j\right) t_{N,j}\Delta_N P_j + P_j$
  - (6) Advance $j = j + 1$
- Return P_j

This algorithm has the advantage of weighting the Bernoulli and Newton increments in a non-arbitrary manner. Yet this comes at a cost, here of an additional optimization problem to be solved, and is likely biased towards the Newton increment as it - intuitively via quadratic convergence - generally takes larger steps, making each increment more likely to be favored over the timid Bernoulli one. This carries again the potential of

losing the advantage of Bernoulli as formulated in the baseline algorithm that guarantees convergence to a particular solvent.

4. Theoretical and Practical Considerations

4.1. Initial Value

All Bernoulli iterations need an initial value, P_0. In contrast to Newton methods, see Meyer-Gohde and Saecker (2022), the baseline Bernoulli method has strong convergence results, see the next subsection, and hence this initial value is of lesser importance. Yet many of the algorithms presented above combine Bernoulli and Newton methods and are subject to this concern. As the goal is to obtain the minimal solvent P, I choose the initial value $P_0 = 0$. In the absence of any other guidance, this choice satisfies the requirement of having all eigenvalues inside the unit circle.

Furthermore, Higham and Kim (2000, p. 512) note that the Bernoulli algorithm can be rerun or restated with a different initialization should numerical difficulties be encountered. For the baseline Bernoulli iteration which solves

$$\left(AP_j + B\right)P_{j+1} = -C \tag{75}$$

the (near) singularity of $AP_j + B$ would pose such a difficulty. In this case the rank deficiency of the leading coefficient matrix admits multiple solutions and I chose the norm ($min \left\| \left(AP_j + B\right)P_{j+1} + C \right\|_F$) minimizing $P_{j+1} = -\left(AP_j + B\right)^+ C$, where $^+$ indicates the Moore-Penrose inverse.

4.2. Convergence

While Higham and Kim (2000) and Bai and Gao (2007) provide convergence results for Bernoulli iterations, their analyses assume that A is nonsingular and that both a minimal and a dominant solvent exist. This is untenable in DSGE models where singular A's abound - associated with variables that arise only at time t and $t-1$ - and singular C's - associated with variables that arise only at time $t+1$ and t - prevent the application of their results to the reverse quadratic. Hence, convergence results for the Bernoulli iterations above for DSGE models are lacking and I provide one in the following.

Recall the baseline Bernoulli method,

$$P_{j+1} = -\left(AP_j + B\right)^{-1} C = F\left(P_j\right) \tag{76}$$

the Fréchet derivative of F at P_j in the direction ΔP_j is $\mathscr{D}_{P_j}F\left(\Delta P_j\right)$ and be defined implicity by differentiating $\left(A P_j + B\right) F\left(P_j\right) = -C$

$$A\Delta P_j F\left(P_j\right) + \left(A P_j + B\right)\mathscr{D}_{P_j}F\left(\Delta P_j\right) = 0 \tag{77}$$

$$\mathscr{D}_{P_j}F\left(\Delta P_j\right) = -\left(A P_j + B\right)^{-1} A\Delta P_j F\left(P_j\right) \tag{78}$$

$$\mathscr{D}_{P_j}F\left(\Delta P_j\right) = \left(A P_j + B\right)^{-1} A\Delta P_j \left(A P_j + B\right)^{-1} C \tag{79}$$

using the Kronecker / vectorized representation ($\text{vec}(ABC) = (C' \otimes A)\text{vec}(B)$)

$$\text{vec}(\mathscr{D}_{P_j}F\left(\Delta P_j\right)) = \left(\left[\left(A P_j + B\right)^{-1} C\right]' \otimes \left[\left(A P_j + B\right)^{-1} A\right]\right)\text{vec}(\Delta P_j) \tag{80}$$

the algorithm converges (locally) to the minimal solvent.

Theorem 1 (Convergence to the unique stable solvent P)**.** *Assume there exists a unique solvent P of $M(P) \equiv A P^2 + B P + C$ in (18) such that the eigenvalues of P comprise all the latent roots, λ of $M(\lambda)$ - see 19 - stable with respect to the closed unit circle. Then the Bernoulli iteration $P_{j+1} = -\left(A P_j + B\right)^{-1} C$ is stable in the neighborhood of P.*

Proof. See the appendix. □

The conditions for the existence of the unique solvent P are Blanchard and Kahn's (1980) celebrated rank and order conditions, see Lan and Meyer-Gohde (2014) and Meyer-Gohde (2022) for the conditions expressed in terms of the general class of multivariate singular leading A models pervasive in the literature today. So conditional on its existence, the Bernoulli method above will converge asymptotially to P.

While the convergence to a specific solvent (in this formulation, the unique stable one) is an advantage over Newton methods, which cannot guarantee convergence to a particular solvent (see Higham and Kim (2001) and Meyer-Gohde and Saecker (2022)), Bernoulli methods converge only at a linear rate (given above by the ratio of the largest stable and smallest unstable eigenvalues) instead of Newton methods' quadratic rate. In practice, I follow Higham and Kim (2001) and use the relative residual $\left\|M(P_j)\right\|_F / \left(\|A\|_F \left\|P_j^2\right\|_F + \|B\|_F \left\|P_j\right\|_F + \|C\|_F\right) < n_y \epsilon$ to assess whether convergence has occurred.

4.3. Accuracy

The practical forward error bounds of Meyer-Gohde (2022) can be used to assess the accuracy of a computed solution $\hat{P}$

$$\underbrace{\frac{\left\|P-\hat{P}\right\|_F}{\left\|P\right\|_F}}_{\text{Forward Error}} \leq \underbrace{\frac{\left\|H_{\hat{P}}^{-1}\text{vec}\left(R_{\hat{P}}\right)\right\|_2}{\left\|\hat{P}\right\|_F}}_{\text{Forward Error Bound 1}} \leq \underbrace{\left\|H_{\hat{P}}^{-1}\right\|_2 \frac{\left\|R_{\hat{P}}\right\|_F}{\left\|\hat{P}\right\|_F}}_{\text{Forward Error Bound 2}} \tag{81}$$

where $R_{\hat{P}} = A\hat{P}^2 + B\hat{P} + C$ is the residual of the matrix quadratic and $H_{\hat{P}} = I_{n_y} \otimes \left(A\hat{P}+B\right) + \hat{P}' \otimes A$. Stewart's (1971) separation function, see also Kågström (1994), Kågström and Poromaa (1996), and Chen and Lv (2018), is

$$\begin{aligned}
\text{sep}\left[\left(A, A\hat{P}+B\right), \left(I, -\hat{P}\right)\right] &= \min_{\left\|X\right\|_F=1} \left\|AX\hat{P} + \left(A\hat{P}+B\right)X\right\|_F && (82)\\
&= \min_{\left\|\text{vec}(X)\right\|_2=1} \left\|H_{\hat{P}}\text{vec}(X)\right\|_2 && (83)\\
&= \sigma_{\min}\left(H_{\hat{P}}\right) \leq \min\left|\lambda\left(A, A\hat{P}+B\right) - \lambda\left(\hat{P}\right)\right| && (84)
\end{aligned}$$

where $\lambda\left(A, A\hat{P}+B\right)$ is the spectrum or set of (generalized) eigenvalues of the pencil $\left(A, A\hat{P}+B\right)$ (and, accordingly, $\lambda\left(\hat{P}\right)$ the set of eigenvalues of $\hat{P}$) and the last line holds with equality for $A = I$ and $\hat{P}$ and $\hat{P}+B$ regular - hence, the separation between the two pencils - the smallest singular value of $H_{\hat{P}}$ - is generically smaller than the minimal separation between their spectra. Analogously to the generalized Sylvester and algebraic Riccati equations, the separation function provides the natural extension of the conditioning number from standard linear equations to these structured problems, and the a posteriori condition number for the matrix quadratic is given by $\text{sep}^{-1}\left[\left(A, A\hat{P}+B\right), \left(I, -\hat{P}\right)\right] = \left\|H_{\hat{P}}^{-1}\right\|_2 = \sigma_{\min}\left(H_{\hat{P}}\right)^{-1}$, which - from above - can be arbitrarily larger than the inverse of the minimal distance between the spectra of the pencils $\left(A, A\hat{P}+B\right), \left(I, -\hat{P}\right)$. This inverse of the separation relates an upper bound to the forward error directly to the residual, like the condition number for a standard linear system, and a tighter bound takes into account the structure more carefully and considers the linear operator $H_{\hat{P}}$ and the residual $R_{\hat{P}}$ jointly.

5. Applications

I conduct two sets of experiments to assess the performance of the Bernoulli algorithms presented above. These two sets are chosen to assess the different methods in a specific, policy relevant model but also in as non-model specific an environment as possible. To

these ends I compare the algorithms above with Dynare's QZ-based method[5] and Newton's method from Meyer-Gohde and Saecker (2022).[6] both in the model of Smets and Wouters (2007) and on the suite of models in the Macroeconomic Model Data Base (MMB) (see Wieland, Cwik, Müller, Schmidt, and Wolters, 2012; Wieland, Afanasyeva, Kuete, and Yoo, 2016), a model comparison initiative at the Institute for Monetary and Financial Stability (IMFS), [7]. The performance is measured in terms of accuracy, computational time, and convergence to the stable solvent, initializing both from zero matrix (an uninformed initialization of a stable solvent) and the output from the QZ algorithm.

5.1. Smets and Wouters's (2007) Model

I begin with the medium scale, estimated model of Smets and Wouters (2007) that is arguably the benchmark for policy analysis. In their model they analyze and estimate a New Keynesian DSGE model with US data featuring the usual frictions, sticky prices and wages, inflation indexation, consumption habit formation as well as production frictions concerning investment, capital and fixed costs. Among the equations is the following log-linearized monetary policy rule that will be the focus of the final experiment, assessing the accuracy of the methods here when solving under alternate, but nearby parameterizations.

$$r_t = \rho r_{t-1} + (1-\rho)(r_\pi \pi_t + r_Y (y_t - y_t^p)) + r_{\Delta y}((y_t - y_t^p) - (y_{t-1} - y_{t-1}^p)) + \varepsilon_t^r, \tag{85}$$

This Taylor rule sets the interest rate r_t according to inflation π_t, the current output gap $(y_t - y_t^p)$ and the change in the output gap, with the parameters r_π, r_Y and $r_{\Delta y}$ describing the strength of each of these reactions and ρ controlling the degree of interest rate smoothing. The monetary policy shock, ε_t^r, follows an AR(1)-process with an iid normal error. The Bayesian estimation of the model employs seven macroeconomic time series from the US economy to estimate the model parameters and the authors show that the model matches the US macroeconomic data very closely and that out-of-sample forecasting performance is favorable compared to (B)VAR models.

[5] See Villemot (2011).

[6] Additionally, note that I follow Dynare and reduce the dimensionality of the problem by grouping variables and structuring the matrix quadratic according to the classification of "static", "purely forward", "purely backward looking", and "mixed" variables. The details are in the online appendix and Meyer-Gohde and Saecker (2022).

[7] See http://www.macromodelbase.com

Method	Relative Performance		Forward Errors		Iterations
	Run Time	Max Abs. Diff.	Bound 1	Bound 2	
Dynare (QZ)	0.00063		5.5e-14	2.4e-11	1
Baseline Newton	2.4	108	4.1e-14	1.6e-09	11
Baseline Bernoulli	12	7.7e-13	3.8e-14	2.6e-11	436
Modified Bernoulli (MBI)	132	8.8e-13	3.5e-14	2.5e-11	423
Bernoulli with Line Searches	20	6.9e-13	3.7e-14	2.4e-11	423
Newton-Bernoulli	8.9	108	2.3e-14	1.5e-09	39
Newton-Bernoulli Column	611	6.9e-13	0.49	3.8e+06	2385
Newton-Bernoulli 1/3	11	6.8e-13	4.7e-15	4.1e-12	47
Newton-Bernoulli LS	8.5	6.7e-13	1.1e-14	2.2e-12	33
Newton-Bernoulli Column LS	9.3	7.2e-13	5.1e-15	9.6e-12	32
Newton-Bernoulli LS 1/3	15	5.9e-13	1.1e-14	5.1e-12	57
Newton-Bernoulli Opt	5.6	108	2.3e-14	1e-09	18
Newton-Bernoulli Opt LS	6.5	7.7e-13	1.5e-15	2.2e-12	19

TABLE 1. Results: Model of Smets and Wouters (2007), Posterior Mode

- For Dynare, refer to Adjemian, Bastani, Juillard, Mihoubi, Perendia, Ratto, and Villemot (2011).
- Run Time for Dynare in seconds, for all others, run time relative to Dynare.
- Max Abs. Diff. measures the largest absolute difference in the computed P of each method from the P produced by Dynare.
- Forward error 1 and 2 are the upper bounds for the true forward error, see (81).

Table 1 summarizes the results at the posterior mode calibration of the model of Smets and Wouters (2007). The baseline Bernoulli method takes 12 times longer than QZ, which would appear to put it at a disadvantage compared to the baseline Newton of Meyer-Gohde and Saecker (2022), however the large maximal absolute difference to the QZ solution of the Newton algorithm shows that it has converged to a different solution than the unique stable solution found by QZ. Indeed, this danger looms with Newton related algorithms as can be seen here for the Newton-Bernoulli and Newton-Bernoulli optimal algorithms, both of which also converged to a different solvent. The baseline Bernoulli required 440 iterations and line searches reduced this number to 420, but the reduction in iterations was outweighed by the costliness of the line search algorithm, resulting altogether in a longer computation time. The columnwise Newton-Bernoulli and the modified Bernoulli both took extraordinarily long times to solve the model, with the

Newton-Bernoulli with line searches and optimal Newton-Bernoulli with line searches providing solutions within an order of magnitude of computation time relative to QZ, 33 and 19 iterations respectively to do so, and providing solutions that are an order of magnitude more accurate than QZ.

Method	Relative Performance		Forward Errors		Iterations
	Run Time	Variance π_t	Bound 1	Bound 2	
Dynare (QZ)	0.0046	0.28	1e-11	4.6	1
Baseline Newton	4.3	0.45	2.4e-14	0.00058	4
Baseline Bernoulli	3	0.39	3.8e-13	0.018	90
Modified Bernoulli (MBI)	2814	—	2.9	6.6e+06	5e+04
Bernoulli with Line Searches	504	—	0.99	1.9e+11	5e+04
Newton-Bernoulli	5.1	0.39	3.3e-15	0.00081	8
Newton-Bernoulli Column	3.7	0.4	3.7e-14	0.019	5
Newton-Bernoulli 1/3	1.2	0.46	1.4e-14	0.0027	8
Newton-Bernoulli LS	7.7	0.39	7.9e-15	0.0023	6
Newton-Bernoulli Column LS	5.7	0.44	1.7e-14	0.00079	6
Newton-Bernoulli LS 1/3	1	0.38	3.3e-14	0.0022	8
Newton-Bernoulli Opt	3.7	0.45	2.4e-14	0.00058	4
Newton-Bernoulli Opt LS	4.2	0.5	1.3e-15	0.0011	4

TABLE 2. Results: Model of Smets and Wouters (2007), Numerically Problematic Parameterization

- For Dynare, refer to Adjemian, Bastani, Juillard, Mihoubi, Perendia, Ratto, and Villemot (2011).
- Run Time for Dynare in seconds, for all others, run time relative to Dynare.
- Variance π_t gives the associated value for the population or theoretical variance of inflation - note that two algorithms did not converge to a stable P and hence the variance could not be calculated for them.
- Forward error 1 and 2 are the upper bounds for the true forward error, see (81).

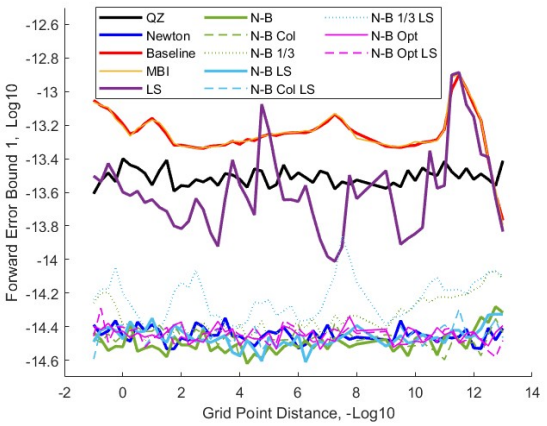
Table 2 assesses the different methods as solution refinement techniques, by parameterizing the model of Smets and Wouters (2007) within the prior to demonstrate an economically relevant numerical instability. The second column now displays the variance of inflation as predicted by the different solution methods,[8] At this parameterization, the QZ-based solution predicts an inflation variance of 0.28. However, even the lower of the

[8]See Meyer-Gohde (2022) for more details on the parameterization. Smets and Wouters (2007) report a variance of inflation in the entire sample of 0.62 and 0.55 and 0.25 in two subsamples.

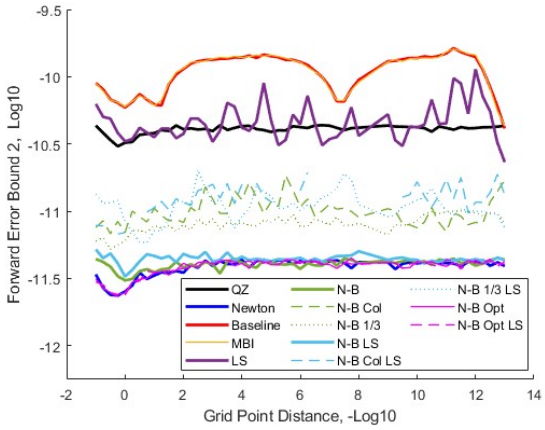
two upper bounds on the forward error is consistent with a numerical instability being several orders of magnitude beyond machine precision.

In the final experiment with the model of Smets and Wouters (2007), I use algorithms to solve iteratively for different parameterizations of the Taylor rule. The goal here is to explore whether solutions from previous, nearby parameterizations can be used to efficiently initialize the Newton methods similarly to the experiment above with the QZ solution as the initial guess. For the parameters determining the Taylor rule reaction to inflation and the long run reaction to the output gap, the experiment iterates through a grid of 10×10 parameter values varying the size of the interval considered - setting $r_\pi \in [1.5, 1.5\,(1+10^{-x})]$ and $r_Y \in [0.125, 0.125\,(1+10^{-x})]$, where $x \in [-1, 8]$ (Smets and Wouters (2007) calibrate them to $r_\pi = 2.0443$ and $r_Y = 0.0882$). The algorithm iterates through the two-dimensional grid taking the solution under the previous parameterization as the initialization for the next iteration. A decrease in the spacing between the 100 grid points thus increases the precision of the starting guess, the solution from the previous parameterization.

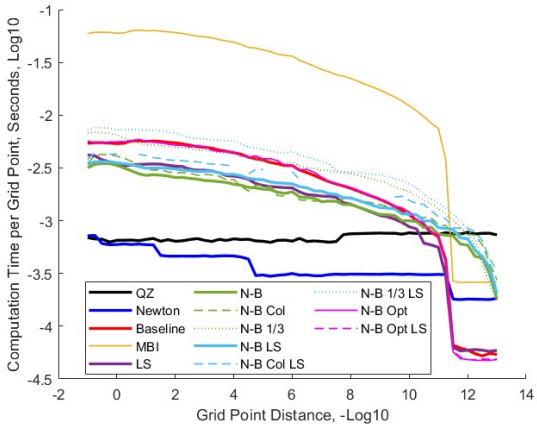
Figure 1 summarizes the experiment graphically. Figure 1c confirms a decrease in run time per grid point with a narrower grid for the iterative algorithms here and an irrelevance of the grid spacing for QZ. As the grid becomes narrower, the iterative Bernoulli and Newton procedures increasingly benefit from starting from the solution of the previous iteration as it becomes closer to the unknown solution of the current iteration. The QZ algorithm does not operate iteratively and, hence, demonstrates no such benefit, solving for each grid point anew. The baseline Newton algorithm of Meyer-Gohde and Saecker (2022) moves in a clear step like fashion - i.e., as the number of iterations it requires is few, a saved iteration is visually apparent. All of the Bernoulli algorithms demonstrate the same pattern, fewer iterations are needed as the grid becomes tighter and, when only one iteration is needed, the algorithms take a large step down in computation costs, with the baseline, line search, and optimal algorithms significantly less costly for tight enough grids than both QZ and the baseline Newton method. According to figures 1a, 1b, overall, all algorithms involving a Newton step are significantly more precise, the line search equivalently precise, and the baseline and modified Bernoulli somewhat less precise than QZ.



(A) Forward Error 1



(B) Forward Error 2



(C) Computation Time per Grid Point

FIGURE 1. Forward Errors and Computation Time per Grid Point for different parameterizations of the model by Smets and Wouters (2007).

Figures 1a, 1b plot the upper forward error bounds 1 and 2 against the grid size, log10 scale on both axes. Figure 1c plots the computation per grid point against the number of grid points, log10 scale on both axes.

5.2. MMB Suite Comparison

The results above are inherently model specific. While potentially indicative, it is unclear what to expect in other settings. To attenuate this, I turn to the Macroeconomic Model Data Base (MMB) (see Wieland, Cwik, Müller, Schmidt, and Wolters, 2012; Wieland, Afanasyeva, Kuete, and Yoo, 2016),a model comparison initiative at the Institute for Monetary and Financial Stability (IMFS)[9] traditionally used to compare the predicted outcomes of different policies across a broad set of macroeconomic models. Version 3.1

[9]See http://www.macromodelbase.com.

contains 151 different models, ranging from small scale, pedagogical models to large scale, estimated models of the US, EU, and multi-country economies. While certainly invaluable for exploring the possible outcomes of policy interventions, this database presents the literature with a useful tool for assessing the potential of different solution methods in a more model-robust context than is currently done in the DSGE literature. Accordingly, I apply the methods of this paper to the set of models appropriate for reproduction,[10] the varying sizes of which are summarized in figure 2. Reiterating this point, this is the same suite of models used in Meyer-Gohde and Saecker (2022) and Meyer-Gohde (2022), which facilitates the comparison of the methods.

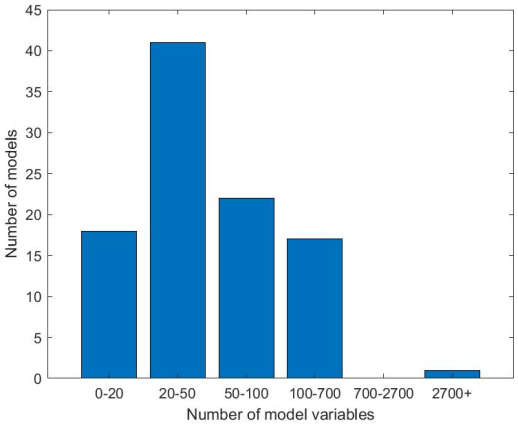


FIGURE 2. Histogram over the number of variables for the 99 MMB models

Figure 2 plots the number of model variables over the amount of MMB models. Currently the total amount of models considered is 99.

Now I examine the remainder of the models of the MMB and compare the various Bernoulli methods to the QZ and Newton methods. I solve each applicable model in the MMB 100 times, initializing the methods with a zero matrix and present the results as the average within the middle three quintiles across runs to reduce the effects of outliers.

Table 3 summarizes the results. The first column of results counts the number of models for which the method in question converged to the unique stable solution, highlighting the well-known (Higham and Kim, 2001; Meyer-Gohde and Saecker, 2022) drawback of Newton methods, namely the unpredictability of which solution the algorithm will converge to. This problem is not faced by the baseline Bernoulli method, as is to be expected following theorem 1 above, which converged for all models in the MMB. Note that all of the algorithms here performed better than the baseline Newton algorithm

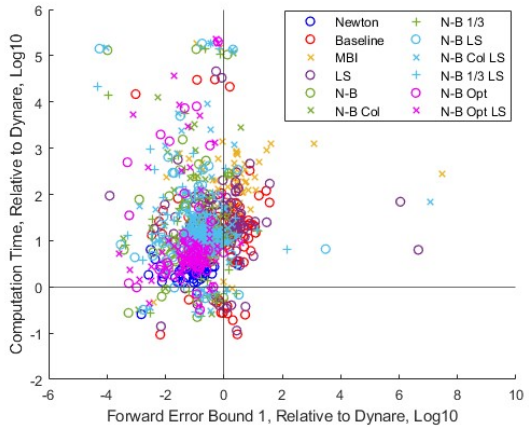
[10]Currently, this is 99 models, ranging from small scale DSGE models to models from policy institutions containing hundreds of variables. Some of the models in the database are deterministic and/or use nonlinear or non-rational (e.g., adaptive) expectations and, hence, are not appropriate for our comparison here.

Method	Convergence	Run Time			Forward Error 1			Forward Error 2			Iterations
		Median	Min	Max	Median	Min	Max	Median	Min	Max	
Dynare (QZ)	99	1	1	1	1	1	1	1	1	1	1
Baseline Newton	53	2.1	0.26	9.8	0.1	5.7e-17	3	0.087	4.1e-17	1.5	8
Baseline Bernoulli	99	21	0.093	3.1e+04	1.3	0.00096	37	1.1	0.0007	325	545
Modified Bernoulli (MBI)	62	224	0.46	1.8e+05	2.2	0.0039	3.1e+07	1.8	0.011	4.1e+06	650
Bernoulli with Line Searches	95	22	0.11	4.6e+04	1	0.00012	4.6e+06	0.96	6.2e-06	4.9e+06	447
Newton-Bernoulli	84	7.2	0.22	1.4e+05	0.1	0.0001	2.2e+12	0.13	5.1e-06	5.2e+28	25
Newton-Bernoulli Column	87	11	0.26	1.5e+05	0.16	0.00035	8.8e+12	0.22	0.003	1.6e+17	33
Newton-Bernoulli 1/3	95	14	0.23	1.4e+05	0.18	0.00011	14	0.19	4.4e-06	1.5e+02	54
Newton-Bernoulli LS	87	11	0.27	1.8e+05	0.12	5.5e-05	3e+12	0.12	6.2e-06	6.9e+13	27
Newton-Bernoulli Column LS	91	14	0.3	1.8e+05	0.14	9.3e-05	3.3e+10	0.24	3.9e-05	8.6e+16	37
Newton-Bernoulli LS 1/3	93	17	0.29	1.8e+05	0.19	4.7e-05	1.4e+02	0.18	4.4e-06	88	51
Newton-Bernoulli Opt	57	3.7	0.32	2e+05	0.098	0.0005	0.95	0.085	0.00098	1.3	10
Newton-Bernoulli Opt LS	69	5.7	0.4	2.3e+05	0.1	0.00026	3.9	0.083	1e-05	0.99	11

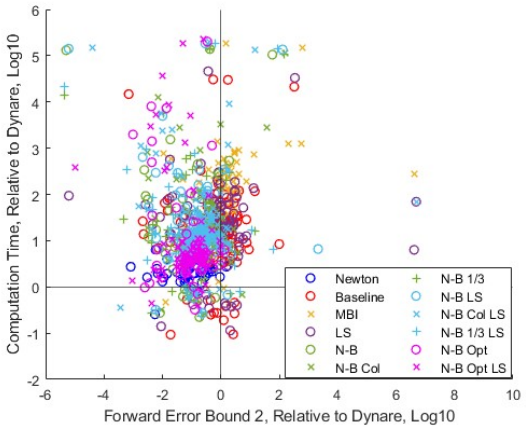
TABLE 3. Results: 99 MMB models (starting guess: zero-matrix).

- For Dynare, refer to Adjemian, Bastani, Juillard, Mihoubi, Perendia, Ratto, and Villemot (2011).
- Run time and forward errors relative to Dynare, number of models converging to the stable solution and median of number of iterations in absolute terms.
- Forward error 1 and 2 are the upper bounds for the true forward error, see (81).

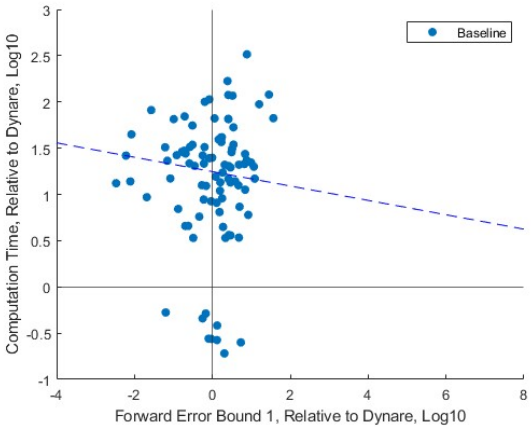
of Meyer-Gohde and Saecker (2022) in this respect. Increasing the weight towards the Bernoulli increment (with $p = 1/3$) increases the number of models for the combined Newton-Bernoulli algorithms converged - both with and without line searches. In general, the accuracy is either comparable or improved within an order of magnitude relative to QZ and the computational time is several to about twenty times larger for all the algorithms - apart from the modified Bernoulli algorithm, which again performed poorly, having the highest computational costs and lowest accuracy.



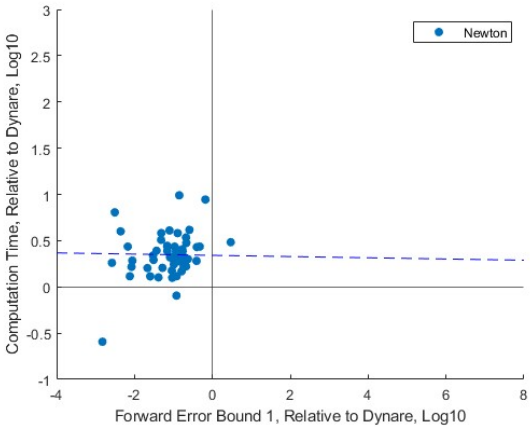
(A) Forward Error 1, All Methods



(B) Forward Error 2, All Methods



(C) Baseline Bernoulli

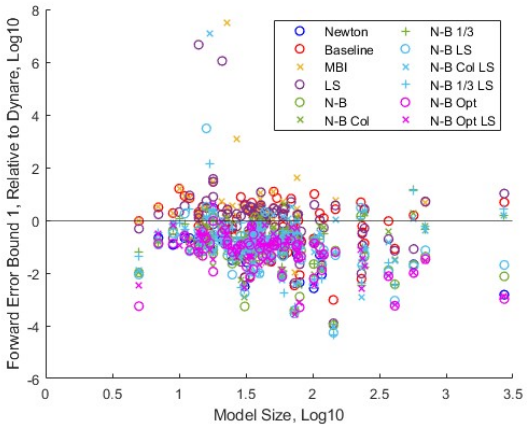


(D) Baseline Newton

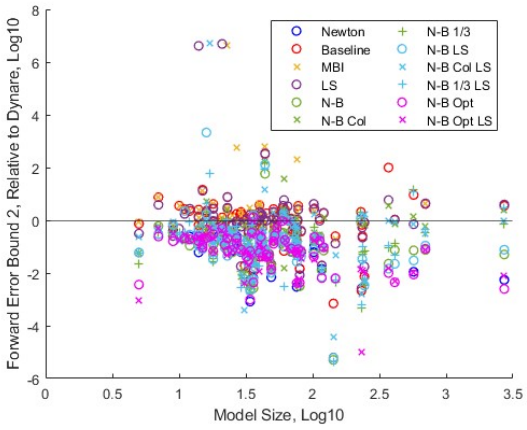
FIGURE 3. Forward Errors and Computation Time Relative to Dynare, log10 scales, for the Macroeconomic Model Data Base (MMB)

Figure 3 provides a model-by-model comparison of the different algorithms' performance relative to QZ. In figures 3a and 3b, the computation times and forward errors (bound 1 and bound 2) relative to QZ are plotted in log10. Hence the cloud of results being primarily in the upper left quadrants leads me to the conclusion that the algorithms are generally more accurate, but computationally more expensive than QZ. In figures 3c and 3d the

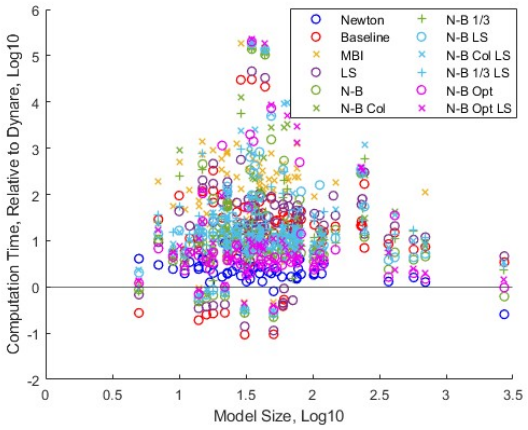
clouds are plotted for the baseline Bernoulli algorithm and Meyer-Gohde and Saecker's (2022) baseline Newton individually using bound 1. Here the conclusion regarding the higher accuracy at marginally higher costs for the Newton algorithm of Meyer-Gohde and Saecker (2022) is apparent. While the cloud for the baseline Bernoulli algorithm straddles the y-axis, implying that it does not systematically provide higher accuracy, the cloud's location above the x-axis indicates that it is computationally more expensive than QZ. The negative trendline indicates, however, a tradeoff, whereby higher accuracy is associated with higher computational costs.



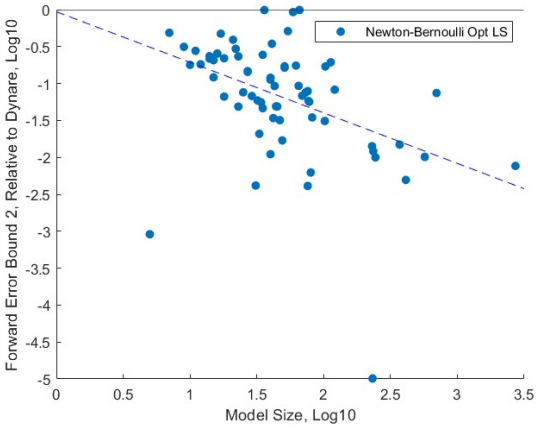
(A) Forward Error 1, Relative to Dynare



(B) Forward Error 2, Relative to Dynare



(C) Computation Time, Relative to Dynare



(D) Forward Error 2, Relative to Dynare

FIGURE 4. Forward Errors, Computation Time and Number of Variables for the Macroeconomic Model Data Base (MMB)

Figures 4a, 4b plot the upper bounds of the forward error 1 and 2 against model size for all methods, log10 scale on both axes.

Figure 4 continues the model-by-model comparison of the different algorithms' performance relative to QZ, but now with a focus on the effect of model size on the algorithms.

In figures 4a and 4b, model sizes - as measured by the number of endogenous variables - and forward errors (bound 1 and bound 2) relative to QZ are plotted in log10. The most striking result is the difference of accuracy, with algorithms involving a Newton step more often below the x axis than the remaining methods, indicating that they are generally associated with more accuracy - an observation I return to shortly in a density comparison. There appears to be a downward trend, indicating that the methods become more accurate relative to QZ for larger models, which is confirmed by looking at figure 4d, which shows the clear downward trend for the optimal Bernoulli-Newton method with line searches and is exemplary for many of the methods. Finally, figure 4c shows that there appears to be a relationship between the size of the model and the computation time relative to Dynare for at least the very large models towards the right of the figure, indicating that the methods here are likely to be particularly competitive alternatives for larger scale applications, with Meyer-Gohde and Saecker's (2022) Newton algorithm presenting the most convincing evidence in this regard.

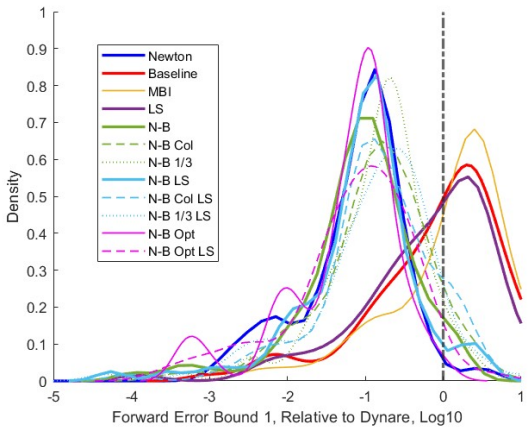
Figure 5 provides an overview of the entire distribution of forward errors, the upper row relative to those from Dynare's QZ method and the lower in absolute terms. Forward errors left of the vertical line are thus smaller than Dynare for both figures in the upper row. For both the first, figure 5a, and second, figure 5b, upper bounds on the forward error, there is an obvious shift to the left of about one order of magnitude for all the methods involving a Newton step and less visually compelling evidence for Bernoulli algorithms not combined with Newton (again, the modified Bernoulli algorithm performs worst). From the lower row, this entails tightening the distributions as well as shifting them closer to machine precision - a lower convergence criterion would allow more iterations and likely bring yet more solutions below machine precision.

To assess the potential for improving on solutions, I repeat the exercise, but now initialize with the solution provided by QZ, see table 4. Here the baseline Bernoulli method is the top performer - with its low per iteration cost, it runs one iteration at a small fraction of the original QZ cost and provides a significant improvement in accuracy. The modified Bernoulli algorithm again performs unsatisfactorily and, interestingly, the combined Bernoulli and Newton methods all require more than one iteration to converge, which would seem to imply that the Newton and Bernoulli steps individually were generally pulling in different directions in the vicinity of the solution provided by QZ. Note that the modified Bernoulli algorithm along with the Newton Bernoulli Column

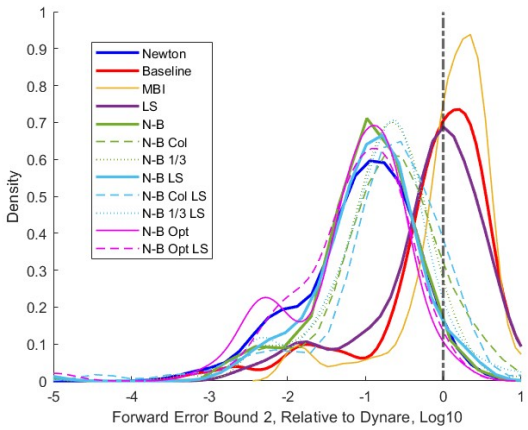
Method	Convergence	Run Time			Forward Error 1			Forward Error 2			Iterations
		Median	Min	Max	Median	Min	Max	Median	Min	Max	
Dynare (QZ)	99	1	1	1	1	1	1	1	1	1	1
Baseline Newton	99	0.971	0.0287	290	0.101	3.49e-15	2.79	0.0932	2.49e-15	1.52	1
Baseline Bernoulli	99	0.174	0.00588	3.49e+03	0.825	0.000275	2.91	0.598	0.000315	214	1
Modified Bernoulli (MBI)	77	1.08	0.052	2.37e+04	0.875	0.000694	3.89	0.705	0.0089	875	1
Bernoulli with Line Searches	99	0.316	0.0224	4.4e+03	0.802	0.000117	3.92	0.578	7.94e-06	299	1
Newton-Bernoulli	99	1.03	0.0512	3.19e+04	0.155	0.000138	1.84	0.15	4.47e-06	91.7	3
Newton-Bernoulli Column	98	1.16	0.0475	3.04e+04	0.205	0.000114	16.2	0.296	0.000126	354	3
Newton-Bernoulli 1/3	99	0.921	0.0333	3.49e+04	0.212	8.66e-05	6.5	0.296	4.69e-06	232	3
Newton-Bernoulli LS	99	1.19	0.0923	3.68e+04	0.147	9.27e-05	2.26	0.168	4.19e-06	184	3
Newton-Bernoulli Column LS	98	1.36	0.0917	3.27e+04	0.159	0.000197	17.4	0.286	0.000128	50.4	3
Newton-Bernoulli LS 1/3	99	1.14	0.0255	3.58e+04	0.207	6.54e-05	2.13	0.289	4.8e-06	169	3
Newton-Bernoulli Opt	99	1.57	0.075	3.71e+04	0.0835	3.49e-15	2.22	0.0851	2.49e-15	1.03	3
Newton-Bernoulli Opt LS	99	1.65	0.0775	3.93e+04	0.0894	8.81e-05	2.79	0.086	5.8e-06	1.03	3

TABLE 4. Results: 99 MMB models (starting guess: solution Dynare (QZ)).

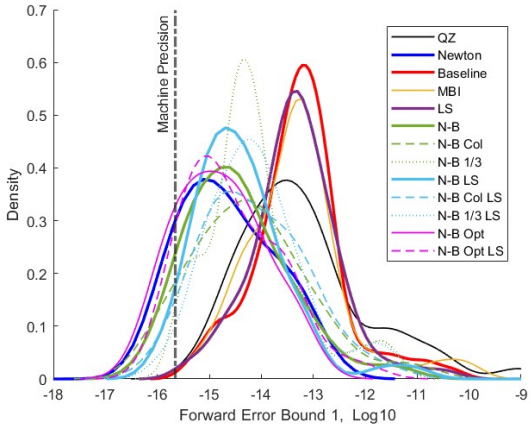
- For Dynare, refer to Adjemian, Bastani, Juillard, Mihoubi, Perendia, Ratto, and Villemot (2011).
- Run Time and foward errors relative to Dynare.
- Forward error 1 and 2 are the upper bounds for the true forward error, see (81).



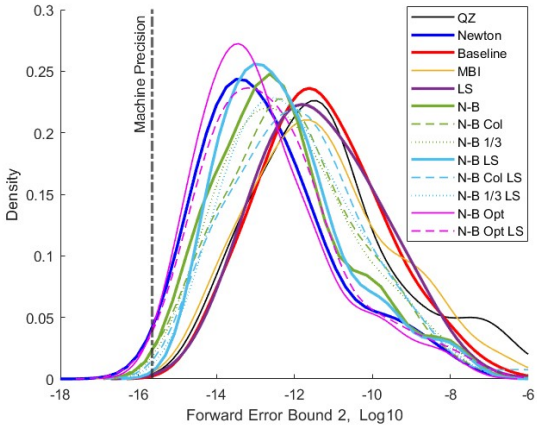
(A) Forward Error 1, Relative to Dynare



(B) Forward Error 2, Relative to Dynare



(C) Forward Error 1



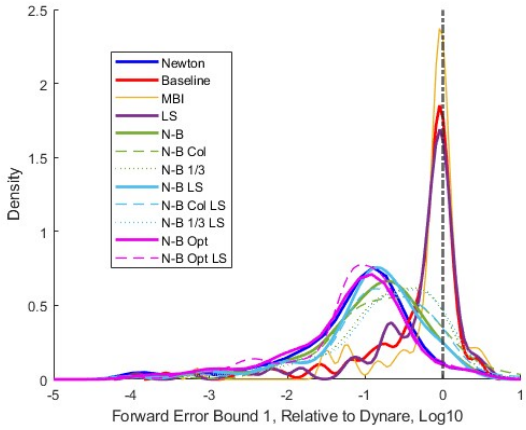
(D) Forward Error 2

FIGURE 5. Distribution of forward error bounds relative to Dynare for the Macroeconomic Model Data Base (MMB)

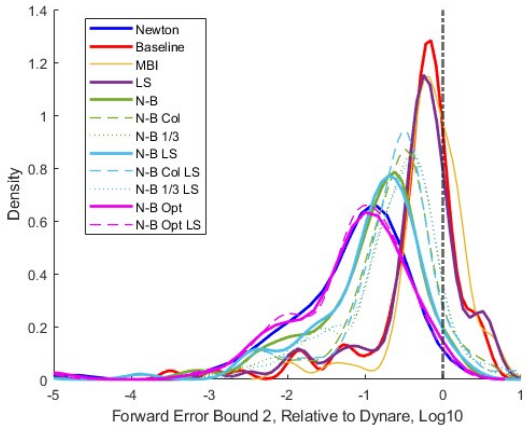
Figures 5a, 5b plot the distribution of model solutions against the upper bounds of the forward error 1 and 2 for all algorithms, log10 scale on the x axis, 99 MMB models (starting guess: zero matrix).

algorithms did not always converge - all three of these algorithms operate column-wise on the problem which apparently can interfere with the convergence, even when initialized close to the solution.

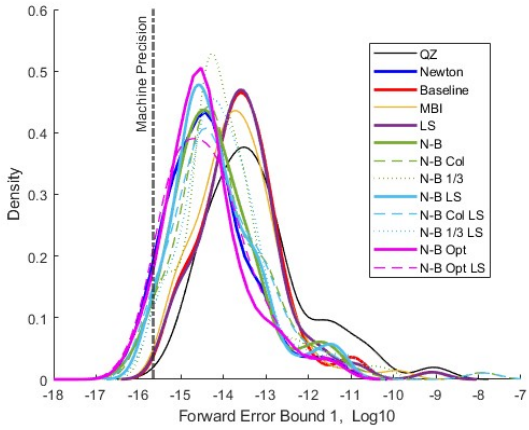
Figure 6, like figure 5 but now initialized at the QZ solution, provides an overview of the entire distribution of forward errors, the upper row relative to those from Dynare's QZ method and the lower in absolute terms. Forward errors left of the vertical line are thus smaller than Dynare for both figures in the upper row. For both the first, figure 6a, and second, figure 6b, upper bounds on the forward error, there is again an obvious shift to the left of about one order of magnitude for all the methods involving a Newton step and a



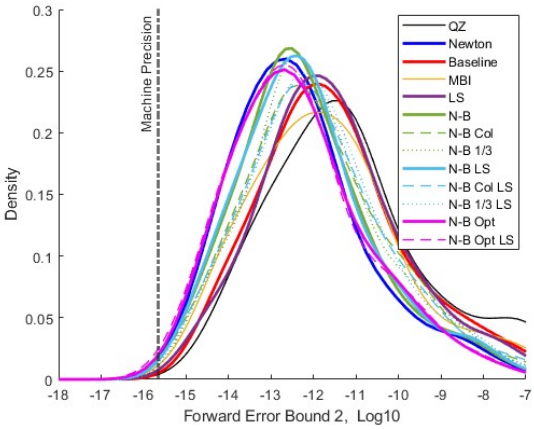
(A) Forward Error 1, Relative to Dynare



(B) Forward Error 2, Relative to Dynare



(C) Forward Error 1



(D) Forward Error 2

FIGURE 6. Distribution of forward error bounds relative to Dynare for the Macroeconomic Model Data Base (MMB)

Figures 6a, 6b plot the distribution of model solutions against the upper bounds of the forward error 1 and 2 for all algorithms, log10 scale on the x axis, 99 MMB models (starting guess: solution Dynare(QZ)).

marginal at best improvement using algorithms without a Newton step. This is consistent with the quadratic convergence properties of Newton methods, see Meyer-Gohde and Saecker (2022), when close to a solution. The modified Bernoulli algorithm now performs comparably to the other methods without a Newton step, highlighting some applicability of the conclusions of Bai and Gao (2007) to the DSGE context.

6. CONCLUSION

I have applied and extended Bernoulli-based methods for solving the matrix quadratic equation underlying the solution of linear DSGE models. This adds a set of alternatives alongside Meyer-Gohde and Saecker's (2022) Newton-based algorithms to the current

standard of a generalized Schur or QZ decomposition (Moler and Stewart, 1973; Golub and van Loan, 2013). Applying the methods to the suite of models in the Macroeconomic Model Data Base (MMB), I demonstrate that Bernoulli-based methods are a potential alternative, with a tradeoff between guaranteed convergence to a particular solution (here the unique, stable solvent) and performance in terms of computational costs.

Particularly in iterative environments or when a solution refinement is sought do these algorithms show potential for future application. In filling in an increasingly dense grid of parameterizations for the Taylor rule in the model of Smets and Wouters (2007), iterative methods like the Bernoulli-based methods here can initialize with the solution from the previous parameterization and significantly outperform the current generalized Schur or QZ method both in terms of computational costs and forward error. Taking the solution from QZ as the initialization, the methods provide roughly an order of magnitude improvement in the accuracy of the solution at a fraction of the original computational cost. This initialization and iteration makes applying the set of Bernoulli methods to improving the accuracy of solutions to linear DSGE models a potentially useful direction of application, as is done in Meyer-Gohde (2022) where QZ based methods from the literature are shown to generate inaccuracies of economic consequence in several macro-finance models.

Future research might explore the application of the methods here to reduce the computational burden associated with solving the model for iterative estimation procedures and might be adapted to more quickly and/or accurately perform likelihood calculations or solve heterogenous agent models.

References

Adjemian, S., H. Bastani, M. Juillard, F. Mihoubi, G. Perendia, M. Ratto, and S. Villemot (2011): "Dynare: Reference Manual, Version 4," Dynare Working Papers 1, CEPREMAP.

Anderson, G. S., A. Levin, and E. Swanson (2006): "Higher-Order Perturbation Solutions to Dynamic Discrete-Time Rational Expectations Models," Discussion Paper 2006-01, Federal Reserve Bank of San Francisco Working Paper Series.

Anderson, G. S., and G. Moore (1985): "A Linear Algebraic Procedure for Solving Linear Perfect Foresight Models," *Economics Letters*, 17(3), 247–252.

Bai, Z.-Z., and Y.-H. Gao (2007): "Modified Bernoulli iteration methods for quadratic matrix equation," *Journal of Computational Mathematics*, 25(5), 498–511.

BINDER, M., AND M. H. PESARAN (1997): "Multivariate Linear Rational Expectations Models: Characterization of the Nature of the Solutions and Their Fully Recursive Computation," *Econometric Theory*, 13(6), 877–88.

BLANCHARD, O. J. (1979): "Backward and Forward Solutions for Economies with Rational Expectations," *The American Economic Review*, 69(2), 114–118.

BLANCHARD, O. J., AND C. M. KAHN (1980): "The Solution of Linear Difference Models under Rational Expectations," *Econometrica*, 48(5), 1305–1311.

CHEN, X. S., AND P. LV (2018): "On estimating the separation between (A,B) and (C,D) associated with the generalized Sylvester equation $AXD - BXC = E$," *Journal of Computational and Applied Mathematics*, 330, 128–140.

DENNIS, JR., J. E., J. F. TRAUB, AND R. P. WEBER (1976): "The Algebraic Theory of Matrix Polynomials," *SIAM Journal on Numerical Analysis*, 13(6), 831–845.

——— (1978): "Algorithms for Solvents of Matrix Polynomials," *SIAM Journal on Numerical Analysis*, 15(3), 523–533.

GANTMACHER, F. R. (1959): *The Theory of Matrices*, vol. I&II. Chelsea Publishing Company, New York, NY.

GOLUB, G. H., AND C. F. VAN LOAN (2013): *Matrix Computations*. The Johns Hopkins University Press, fourth edn.

HAMMARLING, S., C. J. MUNRO, AND F. TISSEUR (2013): "An Algorithm for the Complete Solution of Quadratic Eigenvalue Problems," *ACM Transactions On Mathematical Software*, 39(3), 18:1–18:19.

HIGHAM, N. J. (2002): *Accuracy and Stability of Numerical Algorithms*. Society for Industrial and Applied Mathematics, second edn.

——— (2008): *Functions of Matrices: Theory and Computation*. Society for Industrial and Applied Mathematics.

HIGHAM, N. J., AND H.-M. KIM (2000): "Numerical Analysis of a Quadratic Matrix Equation," *IMA Journal of Numerical Analysis*, 20, 499–519.

——— (2001): "Solving a Quadratic Matrix Equation by Newton's Method with Exact Line Searches," *SIAM Journal on Matrix Analysis and Applications*, 23(2), 499–519.

HORN, R. A., AND C. R. C. R. JOHNSON (2012): *Matrix Analysis*. Cambridge University Press, Cambridge, UK, second edn.

JUDD, K. L. (1992): "Projection Methods for Solving Aggregate Growth Models," *Journal of Economic Theory*, 58(2), 410–452.

——— (1998): *Numerical Methods in Economics*. MIT Press, Cambridge, MA.

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.

KÅGSTRÖM, B. (1994): "A Perturbation Analysis of the Generalized Sylvester Equation $(AR-LB, DR-LE) = (C, F)$," *SIAM Journal on Matrix Analysis and Applications*, 15(4), 1045–1060.

KÅGSTRÖM, B., AND P. POROMAA (1996): "LAPACK-style algorithms and software for solving the generalized Sylvester equation and estimating the separation between regular matrix pairs," *ACM Transactions on Mathematical Software (TOMS)*, 22(1), 78–103.

LAN, H., AND A. MEYER-GOHDE (2014): "Solvability of Perturbation Solutions in DSGE Models," *Journal of Economic Dynamics and Control*, 45, 366–388.

MEYER-GOHDE, A. (2022): "Backward Error and Condition Number Analysis of Linear DSGE Solutions," mimeo, Goethe University Frankfurt, Institute for Monetary and Financial Stability (IMFS).

MEYER-GOHDE, A., AND J. SAECKER (2022): "Solving Linear DSGE Models with Newton Methods," IMFS Working Paper Series 174, Goethe University Frankfurt, Institute for Monetary and Financial Stability (IMFS).

MOLER, C. B., AND G. W. STEWART (1973): "An Algorithm for Generalized Matrix Eigenvalue Problems," *SIAM Journal on Numerical Analysis*, 10(2), 241–256.

SIMS, C. A. (2001): "Solving Linear Rational Expectations Models," *Computational Economics*, 20(1-2), 1–20.

SMETS, F., AND R. WOUTERS (2007): "Shocks and Frictions in US Business Cycles: A Bayesian DSGE Approach," *The American Economic Review*, 97(3), 586–606.

STEWART, G. W. (1971): "Error Bounds for Approximate Invariant Subspaces of Closed Linear Operators," *SIAM Journal on Numerical Analysis*, 8(4), 796–808.

TISSEUR, F., AND K. MEERBERGEN (2001): "The Quadratic Eigenvalue Problem," *SIAM Review*, 43(2), 235–286.

UHLIG, H. (1999): "A Toolkit for Analysing Nonlinear Dynamic Stochastic Models Easily," in *Computational Methods for the Study of Dynamic Economies*, ed. by R. Marimon, and A. Scott, chap. 3, pp. 30–61. Oxford University Press.

VILLEMOT, S. (2011): "Solving Rational Expectations Models at First Order: What Dynare Does," Dynare Working Papers 2, CEPREMAP.

WIELAND, V., E. AFANASYEVA, M. KUETE, AND J. YOO (2016): "New Methods for Macro-Financial Model Comparison and Policy Analysis," in *Handbook of Macroeconomics*, ed. by J. B. Taylor, and H. Uhlig, vol. 2 of *Handbook of Macroeconomics*, pp. 1241–1319. Elsevier.

WIELAND, V., T. CWIK, G. J. MÜLLER, S. SCHMIDT, AND M. WOLTERS (2012): "A new comparative approach to macroeconomic modeling and policy analysis," *Journal of Economic Behavior & Organization*, 83(3), 523–541.

APPENDIX

6.1. Detailed Dynare Topology

Here I summarize the details in the matrix quadratic that follows from the typology of variables from Dynare as laid out in Villemot (2011). See Meyer-Gohde and Saecker (2022) for details.

Subdividing the system of equations in accordance with the QR decomposition yields

$$\underbrace{\begin{array}{c|ccc} & n^s & n^{--} & n^m \\ n^s & \mathbf{0} & \mathbf{0} & \underset{n^s\times n^+}{\check{\mathbf{A}}^+} \\ n^{--} & \mathbf{0} & \mathbf{0} & \\ n^m & \mathbf{0} & \mathbf{0} & \underset{n^d\times n^+}{\tilde{\mathbf{A}}^+} \\ n^{++} & \mathbf{0} & \mathbf{0} & \end{array}}_{A\ n\times n}\underset{n\times n}{\mathbf{P}^2}+\underbrace{\begin{array}{c|ccc} & n^s & n^{--} & n^m \\ n^s & \check{\mathbf{A}}^{0s} & & \underset{n^s\times n^d}{\check{\mathbf{A}}^{0d}} \\ n^{--} & \mathbf{0} & & \\ n^m & \mathbf{0} & & \underset{n^d\times n^d}{\tilde{\mathbf{A}}^0} \\ n^{++} & \mathbf{0} & & \end{array}}_{B\ n\times n}\underset{n\times n}{\mathbf{P}}+\underbrace{\begin{array}{c|ccc} & n^s & n^m & n^{++} \\ n^s & \mathbf{0} & \underset{n^s\times n^-}{\check{\mathbf{A}}^-} & \mathbf{0} \\ n^{--} & \mathbf{0} & & \mathbf{0} \\ n^m & \mathbf{0} & \underset{n^d\times n^-}{\tilde{\mathbf{A}}^-} & \mathbf{0} \\ n^{++} & \mathbf{0} & & \mathbf{0} \end{array}}_{C\ n\times n}=\underset{n\times n}{\mathbf{0}}$$

where n^d is the number of dynamic variables, the sum of number of purely backward-looking, n^{--}, mixed n^m, and purely forward-looking variables, n^{++}. The number of forward-looking variables, n^+, is the sum of the number of mixed, n^m, and purely forward-looking variables, n^{++}, and the number of backward-looking variables, n^-, is the sum of the number of purely backward-looking, n^{--} and mixed variables n^m. Hence, the number of endogenous variables is the sum of the number of static, n^s, and dynamic variables, n^d, or the sum of the number of static, n^s, purely backward-looking, n^{--}, mixed n^m, and purely forward-looking variables, n^{++}. The dimensions satisfy the following

$$n^d = n^{--} + n^m + n^{++}, \quad n^+ = n^m + n^{++}, \quad n^- = n^{--} + n^m, \quad n = n^s + n^d = n^s + n^{--} + n^m + n^{++}$$

The transition matrix, P, from (4) that solves the matrix equation (18) can be subdivided in accordance to Dynare's typology as

$$\mathbf{P} = \begin{array}{c} \\ n^{s} \\ n^{--} \\ n^{m} \\ n^{++} \end{array} \begin{array}{c} \begin{array}{cccc} n^{s} & n^{--} & n^{m} & n^{++} \end{array} \\ \left[\begin{array}{cccc} \mathbf{P}_{s,s} & \mathbf{P}_{s,--} & \mathbf{P}_{s,m} & \mathbf{P}_{s,++} \\ \mathbf{P}_{--,s} & \mathbf{P}_{--,--} & \mathbf{P}_{--,m} & \mathbf{P}_{--,++} \\ \mathbf{P}_{m,s} & \mathbf{P}_{m,--} & \mathbf{P}_{m,m} & \mathbf{P}_{m,++} \\ \mathbf{P}_{++,s} & \mathbf{P}_{++,--} & \mathbf{P}_{++,m} & \mathbf{P}_{++,++} \end{array}\right] \end{array} = \begin{array}{c} n \\ \left[\begin{array}{cccc} \mathbf{P}_{\bullet,s} & \mathbf{P}_{\bullet,--} & \mathbf{P}_{\bullet,m} & \mathbf{P}_{\bullet,++} \end{array}\right] \end{array} = \begin{array}{c} \\ n^{s} \\ n^{--} \\ n^{m} \\ n^{++} \end{array} \begin{array}{c} n \\ \left[\begin{array}{c} \mathbf{P}_{s,\bullet} \\ \mathbf{P}_{--,\bullet} \\ \mathbf{P}_{m,\bullet} \\ \mathbf{P}_{++,\bullet} \end{array}\right] \end{array}$$

The matrix quadratic can be expressed as

$$\begin{aligned} \underset{n\times n}{\mathbf{M}}(\underset{n\times n}{\mathbf{P}}) &= \underset{n\times n}{\mathbf{A}}\,\mathbf{P}^{2} + \underset{n\times n}{\mathbf{B}}\,\mathbf{P} + \underset{n\times n}{\mathbf{C}} \\ &= \underbrace{(\mathbf{AP}+\mathbf{B})}_{\equiv\mathbf{G}}\mathbf{P} + \mathbf{C} \end{aligned}$$

For a solvent P of the matrix quadratic, taking the structure of C from the Dynare typology above into account yields

$$\mathbf{M}(\mathbf{P}) = 0 = \mathbf{GP} + \mathbf{C}$$

$$= \mathbf{G} \begin{array}{c} n \\ \left[\begin{array}{cccc} \mathbf{P}_{\bullet,s} & \mathbf{P}_{\bullet,--} & \mathbf{P}_{\bullet,m} & \mathbf{P}_{\bullet,++} \end{array}\right] \end{array} + \begin{array}{c} \\ n^{s} \\ n^{--} \\ n^{m} \\ n^{++} \end{array} \begin{array}{c} \begin{array}{cccc} n^{s} & n^{--} & n^{m} & n^{++} \end{array} \\ \left[\begin{array}{cccc} \mathbf{0} & \underset{n^{s}\times n^{-}}{\breve{\mathbf{A}}^{-}} & \mathbf{0} \\ \mathbf{0} & & \mathbf{0} \\ \mathbf{0} & \underset{n^{d}\times n^{-}}{\tilde{\mathbf{A}}^{-}} & \mathbf{0} \\ \mathbf{0} & & \mathbf{0} \end{array}\right] \end{array}$$

Following Meyer-Gohde and Saecker (2022), who apply corollary 4.5 of Lan and Meyer-Gohde (2014), if P is the unique solvent of $M(P)$ stable with respect to the closed unit circle, $\mathbf{G}$ has full rank and hence the columns of P associated with nonzero columns in C, the static and forward-looking variables are zero $\rightarrow \mathbf{P}_{\bullet,s} = \underset{n\times n^{s}}{\mathbf{0}}$, $\mathbf{P}_{\bullet,++} = \underset{n\times n^{++}}{\mathbf{0}}$, whence $\mathbf{P}$ is $\mathbf{P} = \begin{array}{c} n \\ \left[\begin{array}{cccc} \mathbf{0} & \mathbf{P}_{\bullet,--} & \mathbf{P}_{\bullet,m} & \mathbf{0} \end{array}\right] \end{array}$ and $\mathbf{M}(\mathbf{P}) = \left[\begin{array}{cccc} \underset{n\times n^{s}}{\mathbf{0}} & \underset{n\times n^{--}}{\mathbf{M}(\mathbf{P})^{--}} & \underset{n\times n^{m}}{\mathbf{M}(\mathbf{P})^{m}} & \underset{n\times n^{++}}{\mathbf{0}} \end{array}\right]$. Consequentially, the first n^{s} rows of the matrix quadratic, taking $\begin{array}{c} n \\ n^{--} \\ n^{m} \\ n^{++} \end{array} \begin{array}{c} \\ \left[\begin{array}{c} \mathbf{P}_{--,\bullet} \\ \mathbf{P}_{m,\bullet} \\ \mathbf{P}_{++,\bullet} \end{array}\right] \end{array}$ as given, yield $\begin{array}{c} n^{--} \quad n^{m} \\ n^{s} \left[\begin{array}{cc} \mathbf{P}_{s,--} & \mathbf{P}_{s,m} \end{array}\right] \end{array}$ as

$$\begin{array}{c} n^{--} \quad n^{m} \\ n^{s} \left[\begin{array}{cc} \mathbf{P}_{s,--} & \mathbf{P}_{s,m} \end{array}\right] \end{array} = -\left[\underset{n^{s}\times n^{s}}{\breve{\mathbf{A}}^{0s}}\right]^{-1} \left(\underset{n^{s}\times n^{+}}{\breve{\mathbf{A}}^{+}} \begin{array}{c} \\ n^{m} \\ n^{++} \end{array} \begin{array}{c} \begin{array}{cc} n^{--} & n^{m} \end{array} \\ \left[\begin{array}{cc} \mathbf{P}_{m,--} & \mathbf{P}_{m,m} \\ \mathbf{P}_{++,--} & \mathbf{P}_{++,m} \end{array}\right] \end{array} \begin{array}{c} \\ n^{--} \\ n^{m} \end{array} \begin{array}{c} \begin{array}{cc} n^{--} & n^{m} \end{array} \\ \left[\begin{array}{cc} \mathbf{P}_{--,--} & \mathbf{P}_{--,m} \\ \mathbf{P}_{m,--} & \mathbf{P}_{m,m} \end{array}\right] \end{array} \right.$$

$$+\underset{n^s\times n^d}{\breve{\mathbf{A}}^{0d}}\begin{array}{c|cc} & n^{--} & n^m \\ \hline n^{--} & \mathbf{P}_{--,--} & \mathbf{P}_{--,m} \\ n^m & \mathbf{P}_{m,--} & \mathbf{P}_{m,m} \\ n^{++} & \mathbf{P}_{++,--} & \mathbf{P}_{++,m} \end{array} + \underset{n^s\times n^-}{\breve{\mathbf{A}}^{-}}\Bigg)$$

and the first n^s rows of P are $\underset{n^s\times n}{\mathbf{P}_{s,\bullet}} = n^s\begin{array}{cccc} n^s & n^{--} & n^m & n^{++} \\ \big[\;\mathbf{0} & \mathbf{P}_{s,--} & \mathbf{P}_{s,m} & \mathbf{0}\;\big] \end{array}.$

The last n^d columns and rows of P solve the reduced matrix quadratic equation

$$\begin{array}{c|ccc} & n^{--} & n^m & n^{++} \\ \hline n^{--} & \mathbf{0} & & \\ n^m & \mathbf{0} & \underset{n^d\times n^+}{\tilde{\mathbf{A}}^{+}} & \\ n^{++} & \mathbf{0} & & \end{array} \; \underbrace{\begin{array}{c|ccc} & n^{--} & n^m & n^{++} \\ \hline n^{--} & \mathbf{P}_{--,--} & \mathbf{P}_{--,m} & \mathbf{P}_{--,++} \\ n^m & \mathbf{P}_{m,--} & \mathbf{P}_{m,m} & \mathbf{P}_{m,++} \\ n^{++} & \mathbf{P}_{++,--} & \mathbf{P}_{++,m} & \mathbf{P}_{++,++} \end{array}}_{\underset{n^d\times n^d}{\tilde{\mathbf{P}}}} \cdot \underset{n^d\times n^d}{\tilde{\mathbf{P}}} + \underset{n^d\times n^d}{\tilde{\mathbf{A}}^{0}}\underset{n^d\times n^d}{\tilde{\mathbf{P}}}$$

$$+\begin{array}{c|ccc} & n^{--} & n^m & n^{++} \\ \hline n^{--} & & & \mathbf{0} \\ n^m & & \underset{n^d\times n^-}{\tilde{\mathbf{A}}^{-}} & \mathbf{0} \\ n^{++} & & & \mathbf{0} \end{array}$$

$$=\underset{n^d\times n^d}{\tilde{\mathbf{M}}(\tilde{\mathbf{P}})} = n^d\begin{array}{ccc} n^{--} & n^m & n^{++} \\ \big[\tilde{\mathbf{M}}(\tilde{\mathbf{P}})^{--} & \tilde{\mathbf{M}}(\tilde{\mathbf{P}})^{m} & \mathbf{0}\big] \end{array} = \underset{n^d\times n^d}{\mathbf{0}}$$

Recalling that $\mathbf{P}_{\bullet,++} = \underset{n\times n^{++}}{\mathbf{0}}$, $\tilde{\mathbf{P}}$ can be reduced and two submatrices $\overline{\mathbf{P}}$ and $\widehat{\mathbf{P}}$ defined via

$$\tilde{\mathbf{P}} = \begin{array}{c|ccc} & n^{--} & n^m & n^{++} \\ \hline n^{--} & \mathbf{P}_{--,--} & \mathbf{P}_{--,m} & \mathbf{P}_{--,++} \\ n^m & \mathbf{P}_{m,--} & \mathbf{P}_{m,m} & \mathbf{P}_{m,++} \\ n^{++} & \mathbf{P}_{++,--} & \mathbf{P}_{++,m} & \mathbf{P}_{++,++} \end{array} = \begin{array}{c|ccc} & n^{--} & n^m & n^{++} \\ \hline n^{--} & \mathbf{P}_{--,--} & \mathbf{P}_{--,m} & \mathbf{0} \\ n^m & \mathbf{P}_{m,--} & \mathbf{P}_{m,m} & \mathbf{0} \\ n^{++} & \mathbf{P}_{++,--} & \mathbf{P}_{++,m} & \mathbf{0} \end{array} \equiv \begin{array}{c|cc} & n^{--}\;\; n^m & n^{++} \\ \hline n^{--} & \underset{n^{--}\times n^-}{\overline{\mathbf{P}}} & \mathbf{0} \\ n^m & \underset{n^+\times n^-}{\widehat{\mathbf{P}}} & \mathbf{0} \\ n^{++} & & \mathbf{0} \end{array}$$

where $\underset{n^{--}\times n^-}{\overline{\mathbf{P}}} \equiv n^{--}\begin{array}{cc} n^{--} & n^m \\ \big[\mathbf{P}_{--,--} & \mathbf{P}_{--,m}\big] \end{array}$ and $\underset{n^+\times n^-}{\widehat{\mathbf{P}}} \equiv \begin{array}{c|cc} & n^{--} & n^m \\ \hline n^m & \mathbf{P}_{m,--} & \mathbf{P}_{m,m} \\ n^{++} & \mathbf{P}_{++,--} & \mathbf{P}_{++,m} \end{array}$ allow the matrix quadratic to be written as

$$\left(\begin{array}{c|ccc} & n^{--} & n^m & n^{++} \\ \hline n^{--} & \mathbf{0} & & \\ n^m & \mathbf{0} & \underset{n^d\times n^+}{\tilde{\mathbf{A}}^{+}} & \\ n^{++} & \mathbf{0} & & \end{array} \begin{array}{c|cc} & n^{--}\;\; n^m & n^{++} \\ \hline n^{--} & \underset{n^{--}\times n^-}{\overline{\mathbf{P}}} & \mathbf{0} \\ n^m & \underset{n^+\times n^-}{\widehat{\mathbf{P}}} & \mathbf{0} \\ n^{++} & & \mathbf{0} \end{array} + \underset{n^d\times n^d}{\tilde{\mathbf{A}}^{0}}\right) \begin{array}{c|cc} & n^{--}\;\; n^m & n^{++} \\ \hline n^{--} & \underset{n^{--}\times n^-}{\overline{\mathbf{P}}} & \mathbf{0} \\ n^m & \underset{n^+\times n^-}{\widehat{\mathbf{P}}} & \mathbf{0} \\ n^{++} & & \mathbf{0} \end{array}$$

$$+\underset{n^d\times n^-}{\tilde{\mathbf{A}}^{-}} = \underset{n^d\times n^d}{\tilde{\mathbf{M}}(\tilde{\mathbf{P}})} = n^d\begin{array}{ccc} n^{--} & n^m & n^{++} \\ \big[\tilde{\mathbf{M}}(\tilde{\mathbf{P}})^{--} & \tilde{\mathbf{M}}(\tilde{\mathbf{P}})^{m} & \mathbf{0}\big] \end{array} = \underset{n^d\times n^d}{\mathbf{0}}$$

which can be reduced to

$$\left(\underset{n^d\times n^+}{\tilde{\mathbf{A}}^+}\ \begin{matrix}n^-\\ n^+\end{matrix}\left[\begin{array}{c|c}\widehat{\mathbf{P}} & \mathbf{0}\\ \hline \end{array}\right]_{n^+\times n^-}\ \begin{matrix}n^{++}\\ \end{matrix}\right)+\underset{n^d\times n^d}{\tilde{\mathbf{A}}^0}\begin{matrix}n^-\\ n^{--}\\ n^+\end{matrix}\left[\begin{array}{c}\overline{\mathbf{P}}\\ \hline \widehat{\mathbf{P}}\end{array}\right]+\underset{n^d\times n^-}{\tilde{\mathbf{A}}^-}=\begin{matrix}n^{--} & n^m\\ \end{matrix}\left[\begin{array}{c|c}\tilde{\mathbf{M}}(\tilde{\mathbf{P}})^{--} & \tilde{\mathbf{M}}(\tilde{\mathbf{P}})^m\end{array}\right]=\underset{n^d\times n^-}{\mathbf{0}}$$

This leads to the Bernoulli iteration

$$\begin{matrix}n^-\\ n^{--}\\ n^+\end{matrix}\left[\begin{array}{c}\overline{\mathbf{P}}_j\\ \hline \widehat{\mathbf{P}}_j\end{array}\right]=-\left(\underset{n^d\times n^+}{\tilde{\mathbf{A}}^+}\ \begin{matrix}n^-\\ n^+\end{matrix}\left[\begin{array}{c|c}\widehat{\mathbf{P}}_{j-1} & \mathbf{0}\\ \hline \end{array}\right]_{n^+\times n^-}\ \begin{matrix}n^{++}\\ \end{matrix}\right)+\underset{n^d\times n^d}{\tilde{\mathbf{A}}^0}\Bigg)^{-1}\underset{n^d\times n^-}{\tilde{\mathbf{A}}^-}$$

6.2. Detailed Dynare Topology - Line Search

The line search methods in the text require finding the minimum of the merit function

$$\begin{aligned}\|M(P+x\Delta P)\|_F^2 &= (1-x)^2\|M(P)\|_F^2+x^2\|A\Delta P(x\Delta P+P)\|_F^2\\ &\quad+(1-x)x*\mathrm{tr}(M(P)^*A\Delta P(x\Delta P+P)\\ &\quad+(x\Delta P^*+xP^*)\Delta P^*A^*M(P))\\ &=(1-x)^2\|M(P)\|_F^2+x^2\|A\Delta P(x\Delta P+P)\|_F^2\\ &\quad+(1-x)x^2*\mathrm{tr}(M(P)^*A\Delta P^2+\Delta P^{*2}A^*M(P))\\ &\quad+(1-x)x*\mathrm{tr}(M(P)^*A\Delta P\cdot P+P^*\Delta P^*A^*M(P))\\ &=(1-x)^2\|M(P)\|_F^2+x^4\left\|A\Delta P^2\right\|_F^2+x^2\|A\Delta P\cdot P\|_F^2\\ &\quad+x^3*\mathrm{tr}(\Delta P^{*2}A^*A\Delta P\cdot P+P^*\Delta P^*A^*A\Delta P^2)\\ &\quad+(1-x)x^2*\mathrm{tr}(M(P)^*A\Delta P^2+\Delta P^{*2}A^*M(P))\\ &\quad+(1-x)x*\mathrm{tr}(M(P)^*A\delta P\cdot P+P^*\Delta P^*A^*M(P))\\ &\quad\equiv t(x)\end{aligned}$$

The first order condition for an interior solution requires finding the zeros of the polynomial

$$g'(x)=4\gamma x^3+3(\xi-\sigma)x^2+2\left(\alpha+\beta-\delta+\sigma\right)x+\delta-2\alpha \tag{A1}$$

where $\alpha=\|M(P)\|_F^2$, $\beta=\|Ad\mathbf{P}\cdot P\|_F^2$, $\gamma=\left\|A\left(d\mathbf{P}\right)^2\right\|_F^2$, $\xi=\mathrm{tr}\left(\left(Ad\mathbf{P}\cdot\mathbf{P}\right)^*A\left(d\mathbf{P}\right)^2+\left(A\left(d\mathbf{P}\right)^2\right)^*Ad\mathbf{P}\cdot\mathbf{P}\right)$, $\sigma=\mathrm{tr}\left(M(P)^*A\left(d\mathbf{P}\right)^2+\left(A\left(d\mathbf{P}\right)^2\right)^*M(P)\right)$, and $\delta=\mathrm{tr}\left(M(P)^*Ad\mathbf{P}\cdot\mathbf{P}+\left(Ad\mathbf{P}\cdot\mathbf{P}\right)^*M(P)\right)$, which follows from

$$\begin{aligned}g'(x) &= 2(1-x)(-1)\|M(P)\|_F^2\\ &\quad+2x\|A\Delta P\cdot P\|_F^2+4x^3\left\|A\Delta P^2\right\|_F^2\\ &\quad+3x^2*\mathrm{tr}(\Delta P^{*2}A^*A\Delta P\cdot P+P^*\Delta P^*A^*A\Delta P^2)\\ &\quad+[(1-x)2x+(-1)x^2]*\mathrm{tr}(M(P)^*A\Delta P^2+\Delta P^{*2}A^*M(P))\\ &\quad+[(1-x)x+(-1)x]*\mathrm{tr}(M(P)^*A\Delta P\cdot P+P^*\Delta P^*A^*M(P))\\ &=-2\|M(P)\|_F^2+2\|M(P)\|_F^2x+2\|A\Delta P\cdot P\|_F^2x+4\left\|A\Delta P^2\right\|_F^2x^3\end{aligned}$$

$$+3*\mathrm{tr}(\Delta P^{*2}A^*A\Delta P\cdot P+P^*\Delta P^*A^*A\Delta P^2)x^2$$
$$+\mathrm{tr}(M(P)^*A\Delta P\cdot P+P^*\Delta P^*A^*M(P))-2*\mathrm{tr}(M(P)^*A\Delta P\cdot P+P^*\Delta P^*A^*M(P))x$$
$$+2*\mathrm{tr}(M(P)^*A\Delta P^2+\Delta P^{*2}A^*M(P))x-3*\mathrm{tr}(M(P)^*A\Delta P^2+\Delta P^{*2}A^*M(P))x^2$$
$$=4\left\|A\Delta P^2\right\|_F^2x^3+(3*\mathrm{tr}(\Delta P^{*2}A^*A\Delta P\cdot P+P^*\Delta P^*A^*A\Delta P^2)-3\,\mathrm{tr}(M(P)^*A\Delta P^2+\Delta P^{*2}A^*M(P)))x^2$$
$$+2\|M(P)\|_F^2x+2\|A\Delta P\cdot P\|_F^2x-2*\mathrm{tr}(M(P)^*A\Delta P\cdot P+P^*\Delta P^*A^*M(P))x$$
$$+2*\mathrm{tr}(M(P)^*A\Delta P^2+\Delta P^{2*}A^*M(P))x+\mathrm{tr}(M(P)^*A\Delta P\cdot P+P^*\Delta P^*A^*M(P))-2\|M(P)\|_F^2$$

Using the typology from Dynare and the results above

$$\alpha=||\mathbf{M}(\mathbf{P})||_F^2=\mathrm{tr}\Big(\mathbf{M}(\mathbf{P})^*\mathbf{M}(\mathbf{P})\Big)$$
$$=\mathrm{tr}\left(\begin{array}{c} \\ n^s \\ n^{--} \\ n^m \\ n^{++}\end{array}\overset{n}{\begin{bmatrix}\mathbf{0}\\ \mathbf{M}(\mathbf{P})^*_{--}\\ \mathbf{M}(\mathbf{P})^*_m\\ \mathbf{0}\end{bmatrix}}\; n\begin{bmatrix}\overset{n^s}{\mathbf{0}} & \overset{n^{--}}{\mathbf{M}(\mathbf{P})_{--}} & \overset{n^m}{\mathbf{M}(\mathbf{P})_m} & \overset{n^{++}}{\mathbf{0}}\end{bmatrix}\right)$$
$$=\mathrm{tr}\Big(\mathbf{M}(\mathbf{P})^*_{--}\mathbf{M}(\mathbf{P})_{--}\Big)+\mathrm{tr}\Big(\mathbf{M}(\mathbf{P})^*_m\mathbf{M}(\mathbf{P})_m\Big)$$

$$\mathbf{M}(\underset{n\times n}{\mathbf{P}})=\underset{n\times n}{\mathbf{A}}\,\mathbf{P}^2+\underset{n\times n}{\mathbf{B}}\,\mathbf{P}+\underset{n\times n}{\mathbf{C}}$$
$$=\begin{array}{c} \\ n^s \\ n^{--} \\ n^m \\ n^{++}\end{array}\begin{bmatrix}\overset{n^s}{\mathbf{0}} & \overset{n^{--}}{\mathbf{0}} & \overset{n^m\quad n^{++}}{\underset{n^s\times n^+}{\breve{\mathbf{A}}^+}}\\ \mathbf{0} & \mathbf{0} & \\ \mathbf{0} & \mathbf{0} & \underset{n^d\times n^+}{\tilde{\mathbf{A}}^+}\\ \mathbf{0} & \mathbf{0} & \end{bmatrix} n\begin{bmatrix}\overset{n^s}{\mathbf{0}} & \overset{n^{--}}{\mathbf{PP}_{\bullet,--}} & \overset{n^m}{\mathbf{PP}_{\bullet,m}} & \overset{n^{++}}{\mathbf{0}}\end{bmatrix}$$
$$+\begin{array}{c} \\ n^s \\ n^{--} \\ n^m \\ n^{++}\end{array}\begin{bmatrix}\overset{n^s}{\breve{\mathbf{A}}^{0s}} & \overset{n^{--}\quad n^m\quad n^{++}}{\underset{n^s\times n^d}{\breve{\mathbf{A}}^{0d}}}\\ \mathbf{0} & \\ \mathbf{0} & \underset{n^d\times n^d}{\tilde{\mathbf{A}}^{0}}\\ \mathbf{0} & \end{bmatrix} n\begin{bmatrix}\overset{n^s}{\mathbf{0}} & \overset{n^{--}}{\mathbf{P}_{\bullet,--}} & \overset{n^m}{\mathbf{P}_{\bullet,m}} & \overset{n^{++}}{\mathbf{0}}\end{bmatrix}$$
$$+\begin{array}{c} \\ n^s \\ n^{--} \\ n^m \\ n^{++}\end{array}\begin{bmatrix}\overset{n^s}{\mathbf{0}} & \overset{n^{--}\quad n^m}{\underset{n^s\times n^-}{\breve{\mathbf{A}}^-}} & \overset{n^{++}}{\mathbf{0}}\\ \mathbf{0} & & \mathbf{0}\\ \mathbf{0} & \underset{n^d\times n^-}{\tilde{\mathbf{A}}^-} & \mathbf{0}\\ \mathbf{0} & & \mathbf{0}\end{bmatrix}$$

$$= n \begin{bmatrix} \overset{n^s}{\mathbf{0}} & \left| \begin{bmatrix} \overset{n^-}{\mathbf{0}} & \overset{n^-}{\mathbf{0}} & \overset{n^+}{\breve{\mathbf{A}}^+} \\ \mathbf{0} & \mathbf{0} & \\ \mathbf{0} & \mathbf{0} & \tilde{\mathbf{A}}^+ \\ \mathbf{0} & \mathbf{0} & \end{bmatrix} \right. \mathbf{P} \begin{bmatrix} \mathbf{P}_{\bullet,--} & \mathbf{P}_{\bullet,m} \end{bmatrix} & \left| \overset{n^{++}}{\mathbf{0}} \right. \end{bmatrix}$$

$$+ n \begin{bmatrix} \overset{n^s}{\mathbf{0}} & \left| \begin{bmatrix} \breve{\mathbf{A}}^{0s} & \breve{\mathbf{A}}^{0d} \\ \mathbf{0} & \\ \mathbf{0} & \tilde{\mathbf{A}}^0 \\ \mathbf{0} & \end{bmatrix} \right. \begin{bmatrix} \mathbf{P}_{\bullet,--} & \mathbf{P}_{\bullet,m} \end{bmatrix} & \left| \overset{n^{++}}{\mathbf{0}} \right. \end{bmatrix} + n \begin{bmatrix} \overset{n^s}{\mathbf{0}} & \left| \begin{bmatrix} \breve{\mathbf{A}}^- \\ \tilde{\mathbf{A}}^- \end{bmatrix} \right. & \left| \overset{n^{++}}{\mathbf{0}} \right. \end{bmatrix}$$

$$\begin{bmatrix} \mathbf{0} & \mathbf{0} & \breve{\mathbf{A}}^+ \\ \mathbf{0} & \mathbf{0} & \\ \mathbf{0} & \mathbf{0} & \tilde{\mathbf{A}}^+ \\ \mathbf{0} & \mathbf{0} & \end{bmatrix} \mathbf{P} \begin{bmatrix} \mathbf{P}_{\bullet,--} & \mathbf{P}_{\bullet,m} \end{bmatrix} = \begin{bmatrix} \breve{\mathbf{A}}^+ \\ \\ \tilde{\mathbf{A}}^+ \end{bmatrix} \mathbf{P}_{m/++,\bullet} \mathbf{P}_{\bullet,--/m}$$

where

$$\mathbf{P}_{\bullet,--/m} = n \begin{bmatrix} \overset{n^{--}}{\mathbf{P}_{\bullet,--}} & \overset{n^m}{\mathbf{P}_{\bullet,m}} \end{bmatrix} \quad \text{and} \quad \mathbf{P}_{\bullet,m/++} = n \begin{bmatrix} \overset{n^m}{\mathbf{P}_{\bullet,m}} & \overset{n^{++}}{\mathbf{P}_{\bullet,++}} \end{bmatrix}$$

$$\mathbf{M}(\mathbf{P}) = n \begin{bmatrix} \overset{n^s}{\mathbf{0}} & \left| \left(\begin{bmatrix} \breve{\mathbf{A}}^+ \\ \\ \tilde{\mathbf{A}}^+ \end{bmatrix} \mathbf{P}_{m/++,\bullet} + \begin{bmatrix} \overset{n^-}{\breve{\mathbf{A}}^{0s}} & \breve{\mathbf{A}}^{0d} \\ \mathbf{0} & \\ \mathbf{0} & \tilde{\mathbf{A}}^0 \\ \mathbf{0} & \end{bmatrix} \right) \right. \mathbf{P}_{\bullet,--/m} + \begin{bmatrix} \breve{\mathbf{A}}^- \\ \tilde{\mathbf{A}}^- \end{bmatrix} & \left| \overset{n^{++}}{\mathbf{0}} \right. \end{bmatrix}$$

$$= n \begin{bmatrix} \overset{n^s}{\mathbf{0}} & \overset{n^-}{\mathbf{X}} & \overset{n^{++}}{\mathbf{0}} \end{bmatrix}$$

where $\mathbf{X}$ is defined as

$$\underset{n \times n^-}{\mathbf{X}} \equiv \begin{matrix} n^s \\ n^d \end{matrix} \overset{n^-}{\begin{bmatrix} \mathbf{X}_1 \\ \mathbf{X}_2 \end{bmatrix}} \equiv \begin{matrix} n^s \\ n^d \end{matrix} \overset{n^-}{\left[\begin{array}{c} \left(\breve{\mathbf{A}}^+_{n^s \times n^+} \mathbf{P}_{m/++,\bullet} + \begin{bmatrix} \breve{\mathbf{A}}^{0s} & \breve{\mathbf{A}}^{0d} \end{bmatrix}_{n^s \times n^s \quad n^s \times n^d} \right) \mathbf{P}_{\bullet,--/m} + \breve{\mathbf{A}}^-_{n^s \times n^-} \\ \hline \tilde{\mathbf{A}}^+_{n^d \times n^+} \mathbf{P}_{m/++,\bullet} \mathbf{P}_{\bullet,--/m} + \begin{bmatrix} \mathbf{0} & \tilde{\mathbf{A}}^0 \end{bmatrix}_{n^d \times n^s \quad n^d \times n^d} \mathbf{P}_{\bullet,--/m} + \tilde{\mathbf{A}}^-_{n^d \times n^-} \end{array} \right]}$$

$$= \begin{matrix} n^s \\ n^d \end{matrix} \overset{n^-}{\left[\begin{array}{c} \left(\breve{\mathbf{A}}^+_{n^s \times n^+} \mathbf{P}_{m/++,\bullet} + \begin{bmatrix} \breve{\mathbf{A}}^{0s} & \breve{\mathbf{A}}^{0d} \end{bmatrix}_{n^s \times n^s \quad n^s \times n^d} \right) \mathbf{P}_{\bullet,--/m} + \breve{\mathbf{A}}^-_{n^s \times n^-} \\ \hline \tilde{\mathbf{A}}^+_{n^d \times n^+} \mathbf{P}_{m/++,\bullet} \mathbf{P}_{\bullet,--/m} + \tilde{\mathbf{A}}^0_{n^d \times n^d} \mathbf{P}_{d,--/m} + \tilde{\mathbf{A}}^-_{n^d \times n^-} \end{array} \right]}$$

$$= \begin{matrix} n^s \\ n^d \end{matrix} \overset{n^-}{\left[\begin{array}{c} \breve{\mathbf{A}}^{0s}_{n^s\times n^s}\mathbf{P}_{s,--/m,\,n^s\times n^-} + \left(\breve{\mathbf{A}}^{+}_{n^s\times n^+}\mathbf{P}_{m/++,d,\,n^+\times n^d} + \breve{\mathbf{A}}^{0d}_{n^s\times n^d}\right)\mathbf{P}_{d,--/m,\,n^d\times n^-} + \breve{\mathbf{A}}^{-}_{n^s\times n^-} \\ \hline \tilde{\mathbf{A}}^{+}_{n^d\times n^+}\mathbf{P}_{m/++,\bullet,\,n^+\times n}\mathbf{P}_{\bullet,--/m,\,n\times n^-} + \tilde{\mathbf{A}}^{0}_{n^d\times n^d}\mathbf{P}_{d,--/m,\,n^d\times n^-} + \tilde{\mathbf{A}}^{-}_{n^d\times n^-} \end{array}\right]}$$

Accordingly

$$\alpha = ||\mathbf{M}(\mathbf{P})||_F^2 = \mathrm{tr}\left(\mathbf{M}(\mathbf{P})^*\mathbf{M}(\mathbf{P})\right) = \mathrm{tr}(\mathbf{X}^*\mathbf{X}) = \mathrm{tr}(\mathbf{X}_1^*\mathbf{X}_1) + \mathrm{tr}(\mathbf{X}_2^*\mathbf{X}_2)$$

by construction $\underset{n^s\times n^-}{\mathbf{X}_1} = \underset{n^s\times n^-}{\mathbf{0}}$ and hence

$$\alpha = ||\mathbf{M}(\mathbf{P})||_F^2 = \mathrm{tr}(\mathbf{X}_2^*\mathbf{X}_2) = \mathrm{tr}\left(\tilde{\mathbf{M}}(\tilde{\mathbf{P}})^*\tilde{\mathbf{M}}(\tilde{\mathbf{P}})\right)$$

Turning to $\gamma = \left|\left|\mathbf{A}d\mathbf{P}^2\right|\right|_F^2$

$$\gamma = \left|\left|\mathbf{A}d\mathbf{P}^2\right|\right|_F^2 = \mathrm{tr}\left(\left(\mathbf{A}d\mathbf{P}^2\right)^*\mathbf{A}d\mathbf{P}^2\right)$$

$$\mathbf{A}d\mathbf{P}^2 = \begin{matrix} n \end{matrix} \overset{\begin{matrix} n^s & n^- & n^{++} \end{matrix}}{\left[\begin{array}{ccc} \mathbf{0} & \mathbf{A}_{n\times n}\,d\,\mathbf{P}_{n\times n}\,d\mathbf{P}_{\bullet,--/m,\,n\times n^-} & \mathbf{0} \end{array}\right]}$$

$$= \begin{matrix} n \end{matrix} \overset{\begin{matrix} n^s & n^- & n^{++} \end{matrix}}{\left[\begin{array}{ccc} \mathbf{0} & \begin{bmatrix} \breve{\mathbf{A}}^{+}_{n^s\times n^+} \\ \tilde{\mathbf{A}}^{+}_{n^d\times n^+} \end{bmatrix} d\mathbf{P}_{m/++,\bullet,\,n^+\times n}\,d\mathbf{P}_{\bullet,--/m,\,n\times n^-} & \mathbf{0} \end{array}\right]}$$

$$= \begin{matrix} n \end{matrix} \overset{\begin{matrix} n^s & n^- & n^{++} \end{matrix}}{\left[\begin{array}{ccc} \mathbf{0} & \begin{bmatrix} \breve{\mathbf{A}}^{+}_{n^s\times n^+} \\ \tilde{\mathbf{A}}^{+}_{n^d\times n^+} \end{bmatrix} d\mathbf{P}_{m/++,--/m,\,n^+\times n^-}\,d\mathbf{P}_{--/m,--/m,\,n^-\times n^-} & \mathbf{0} \end{array}\right]}$$

$$= \begin{matrix} n \end{matrix} \overset{\begin{matrix} n^s & n^- & n^{++} \end{matrix}}{\left[\begin{array}{ccc} \mathbf{0} & \begin{bmatrix} \mathbf{Y}_1 \\ \mathbf{Y}_2 \end{bmatrix} & \mathbf{0} \end{array}\right]}$$

Hence, γ is

$$\gamma = \mathrm{tr}(\mathbf{Y}_1^*\mathbf{Y}_1) + \mathrm{tr}(\mathbf{Y}_2^*\mathbf{Y}_2)$$

By analogy, $\beta = ||\mathbf{A}d\mathbf{P}\cdot\mathbf{P}||_F^2$ is given by

$$\beta = \mathrm{tr}(\mathbf{Z}_1^*\mathbf{Z}_1) + \mathrm{tr}(\mathbf{Z}_2^*\mathbf{Z}_2)$$

where

$$\begin{bmatrix} \mathbf{Z}_1 \\ \mathbf{Z}_2 \end{bmatrix} = \begin{bmatrix} \breve{\mathbf{A}}^{+}_{n^s\times n^+} \\ \tilde{\mathbf{A}}^{+}_{n^d\times n^+} \end{bmatrix} d\mathbf{P}_{m/++,--/m,\,n^+\times n^-}\mathbf{P}_{--/m,--/m,\,n^-\times n^-}$$

Continuing, ξ is given by $\mathrm{tr}\left((Ad\mathbf{P}\cdot\mathbf{P})^* A(d\mathbf{P})^2 + \left(A(d\mathbf{P})^2\right)^* Ad\mathbf{P}\cdot\mathbf{P}\right)$ and as $\mathrm{tr}(A^*B) = \mathrm{tr}(B^*A)$, from

$$(Ad\mathbf{P}\cdot\mathbf{P})^* A(d\mathbf{P})^2 = \begin{matrix} \\ n^s \\ n^d \\ n^{++} \end{matrix}\overset{\begin{matrix} n \end{matrix}}{\begin{bmatrix} \mathbf{0} \\ \begin{bmatrix} \underset{n^-\times n^s}{\mathbf{Y}_1^*} & \underset{n^-\times n^d}{\mathbf{Y}_2^*} \end{bmatrix} \\ \mathbf{0} \end{bmatrix}} \begin{matrix} \\ n \end{matrix}\overset{\begin{matrix} n^s & n^- & n^{++} \end{matrix}}{\begin{bmatrix} \mathbf{0} & \begin{bmatrix} \underset{n^s\times n^-}{\mathbf{Z}_1} \\ \underset{n^d\times n^-}{\mathbf{Z}_2} \end{bmatrix} & \mathbf{0} \end{bmatrix}}$$

follows

$$\mathrm{tr}\left((Ad\mathbf{P}\cdot\mathbf{P})^* A(d\mathbf{P})^2\right) = \mathrm{tr}\left(\mathbf{Y}_1^*\mathbf{Z}_1\right) + \mathrm{tr}\left(\mathbf{Y}_2^*\mathbf{Z}_2\right)$$

Hence,

$$\begin{aligned} \xi &= \mathrm{tr}\left((Ad\mathbf{P}\cdot\mathbf{P})^* A(d\mathbf{P})^2 + \left(A(d\mathbf{P})^2\right)^* Ad\mathbf{P}\cdot\mathbf{P}\right) \\ &= 2\,\mathrm{tr}\left(\mathbf{Y}_1^*\mathbf{Z}_1\right) + 2\,\mathrm{tr}\left(\mathbf{Y}_2^*\mathbf{Z}_2\right) \end{aligned}$$

Finally, $\sigma = \mathrm{tr}\left(M(P)^* A(d\mathbf{P})^2 + \left(A(d\mathbf{P})^2\right)^* M(P)\right)$, and $\delta = \mathrm{tr}\left(M(P)^* Ad\mathbf{P}\cdot\mathbf{P} + (Ad\mathbf{P}\cdot\mathbf{P})^* M(P)\right)$. Using the results from above,

$$\begin{aligned} \mathrm{tr}\left(\mathbf{A}d\mathbf{P}^2\cdot\mathbf{M}(\mathbf{P})^*\right) &= \mathrm{tr}\left(\begin{matrix} \\ n \end{matrix}\overset{\begin{matrix} n^s & n^- & n^{++} \end{matrix}}{\begin{bmatrix} \mathbf{0} & \begin{bmatrix} \underset{n^s\times n^-}{\mathbf{Y}_1} \\ \underset{n^d\times n^-}{\mathbf{Y}_2} \end{bmatrix} & \mathbf{0} \end{bmatrix}} \cdot \begin{matrix} n^s \\ n^- \\ n^{++} \end{matrix}\overset{\begin{matrix} n \end{matrix}}{\begin{bmatrix} \mathbf{0} \\ \begin{bmatrix} \underset{n^-\times n^s}{\mathbf{0}} & \underset{n^-\times n^d}{\tilde{\mathbf{M}}(\tilde{\mathbf{P}})^*} \end{bmatrix} \\ \mathbf{0} \end{bmatrix}}\right) \\ &= \mathrm{tr}\left(\begin{bmatrix} \underset{n^s\times n^-}{\mathbf{Y}_1} \\ \underset{n^d\times n^-}{\mathbf{Y}_2} \end{bmatrix}\begin{bmatrix} \underset{n^-\times n^s}{\mathbf{0}} & \underset{n^-\times n^d}{\tilde{\mathbf{M}}(\tilde{\mathbf{P}})^*} \end{bmatrix}\right) \\ &= \mathrm{tr}\left(\underset{n^d\times n^-}{\mathbf{Y}_2}\ \underset{n^-\times n^d}{\tilde{\mathbf{M}}(\tilde{\mathbf{P}})^*}\right) \end{aligned}$$

and so

$$\begin{aligned} \sigma &= \mathrm{tr}\left(M(P)^* A(d\mathbf{P})^2 + \left(A(d\mathbf{P})^2\right)^* M(P)\right) \\ &= 2\,\mathrm{tr}\left(M(P)^* A(d\mathbf{P})^2\right) = 2\,\mathrm{tr}\left(A(d\mathbf{P})^2 M(P)^*\right) \\ &= 2\,\mathrm{tr}\left(\underset{n^d\times n^-}{\mathbf{Y}_2}\ \underset{n^-\times n^d}{\tilde{\mathbf{M}}(\tilde{\mathbf{P}})^*}\right) \end{aligned}$$

and by analogy

$$\delta = 2\,\mathrm{tr}\left(\underset{n^d\times n^-}{\mathbf{Z}_2}\ \underset{n^-\times n^d}{\tilde{\mathbf{M}}(\tilde{\mathbf{P}})^*}\right)$$

6.3. Detailed Dynare Topology - Combining Bernoulli and Newton

Let $\Delta_B\mathbf{P}_j$ be the Bernoulli increment and $\mathbf{x}_i \in [1,\infty)$ its line-search multiple, and $\Delta_\mathbf{N}\mathbf{P}_j$ be the Newton increment with $\mathbf{y}_i \in (0,2]$, its line search multiple.

The iteration $\mathbf{P}_{j+1} = \mathbf{S}_j\mathbf{x}_j\Delta_\mathbf{B}\mathbf{P}_j + (1-\mathbf{S}_j)\mathbf{y}_j\Delta_j + \mathbf{P}_j$ for $\mathbf{S}_j \in [0,1]$ is a weighted average of the line-search multiples of the Bernoulli and Newton increments. I propose determining S_j according to the same merit

function that was used for $\mathbf{x}_i$, and $\mathbf{y}_i$.

$$\mathbf{t} = \min_{0 \le s \le 1} \|\mathbf{M}(s \cdot \mathbf{x} \cdot \Delta_{\mathbf{B}}\mathbf{P} + (1-s)\mathbf{y}\Delta_{\mathbf{N}}\mathbf{P} + \mathbf{P})\|_F^2$$

Where I have omitted "j" to minizime clutter.

$\mathbf{M}(s \cdot \mathbf{x} \cdot \Delta_{\mathbf{B}}\mathbf{P} + (1-s) \cdot \mathbf{y} \cdot \Delta_N\mathbf{P} + \mathbf{P})$ can be expressed explicitly as

$$\begin{aligned}
\mathbf{M}(s \cdot \mathbf{x} \cdot \Delta_{\mathbf{B}}\mathbf{P} + (1-s) \cdot \mathbf{y} \cdot \Delta_{\mathbf{N}}\mathbf{P} + \mathbf{P}) &= \mathbf{A} \cdot (s \cdot \mathbf{x} \cdot \Delta_{\mathbf{B}}\mathbf{P} + (1-s) \cdot \mathbf{y} \cdot \Delta_{\mathbf{N}} + \mathbf{P})^2 \\
&\quad + \mathbf{B} \cdot (s \cdot \mathbf{x} \cdot \Delta_{\mathbf{B}}\mathbf{P} + (1-s) \cdot \mathbf{y} \cdot \Delta_{\mathbf{N}}\mathbf{P} + \mathbf{P}) + \mathbf{C} \\
&= \mathbf{A}[\mathbf{P}^2 + s \cdot \mathbf{x} \cdot \mathbf{P} \cdot \Delta_{\mathbf{B}}\mathbf{P} + (1-s) \cdot \mathbf{y} \cdot \mathbf{P}\Delta_{\mathbf{N}}\mathbf{P} + s \cdot \mathbf{x} \cdot \Delta_{\mathbf{B}} \cdot \mathbf{P} \cdot \mathbf{P} \\
&\quad + s \cdot (1-s) \cdot \mathbf{x} \cdot \mathbf{y} \cdot \Delta_{\mathbf{B}}\mathbf{P}\Delta_{\mathbf{N}} \cdot \mathbf{P} + s^2 \cdot \mathbf{x}^2 \cdot (\Delta_{\mathbf{B}}\mathbf{P})^2 \\
&\quad + (1-s) \cdot \mathbf{y} \cdot \Delta_{\mathbf{N}}\mathbf{P} \cdot \mathbf{P} \\
&\quad + (1-s) \cdot s \cdot \mathbf{y} \cdot \mathbf{x}\Delta_{\mathbf{N}}\mathbf{P} \cdot \Delta_{\mathbf{B}}\mathbf{P} + (1-s)^2 \cdot \mathbf{y}^2(\Delta_{\mathbf{N}}\mathbf{P})^2] \\
&\quad + \mathbf{B} \cdot (s \cdot \mathbf{x} \cdot \Delta_{\mathbf{B}}\mathbf{P} + (1-s) \cdot \mathbf{y}\Delta_{\mathbf{N}}\mathbf{P} + \mathbf{P}) + \mathbf{C} \\
&= \mathbf{A} \cdot \mathbf{P}^2 + \mathbf{B} \cdot \mathbf{P} + \mathbf{C} \\
&\quad + (1-s) \cdot \mathbf{y} \cdot (\mathbf{A} \cdot \mathbf{P} \cdot \Delta_{\mathbf{N}}\mathbf{P} + \mathbf{A} \cdot \Delta_{\mathbf{N}}\mathbf{P} \cdot \mathbf{P} + \mathbf{B} \cdot \Delta_{\mathbf{N}}\mathbf{P}) \\
&\quad + s \cdot \mathbf{x} \cdot (\mathbf{A} \cdot \mathbf{P} \cdot \Delta_{\mathbf{B}}\mathbf{P} + \mathbf{A} \cdot \Delta_{\mathbf{B}}\mathbf{P} \cdot \mathbf{P} + \mathbf{B}\Delta_{\mathbf{B}}\mathbf{P}) \\
&\quad + \mathbf{A}(s \cdot \mathbf{x} \cdot \Delta_{\mathbf{B}}\mathbf{P} + (1-s) \cdot \mathbf{y} \cdot \Delta_{\mathbf{N}}\mathbf{P})^2
\end{aligned}$$

Note that

$$\begin{aligned}
\mathbf{A}(\mathbf{P} \cdot \Delta_{\mathbf{N}}\mathbf{P} + \Delta_{\mathbf{N}}\mathbf{P} \cdot \mathbf{P}) + \mathbf{B} \cdot \Delta_{\mathbf{N}}\mathbf{P} &= -\mathbf{M}(\mathbf{P}) \\
\mathbf{A}(\mathbf{P} \cdot \Delta_{\mathbf{N}}\mathbf{P} + \Delta_{\mathbf{N}}\mathbf{P} \cdot \mathbf{P}) + \mathbf{B} \cdot \Delta_{\mathbf{N}}\mathbf{P} &= -\mathbf{M}(\mathbf{P}) + \mathbf{A} \cdot \Delta_{\mathbf{B}}\mathbf{P} \cdot \mathbf{P} \\
\mathbf{A}\mathbf{P}^2 + \mathbf{B}\mathbf{P} + \mathbf{C} &= \mathbf{M}(\mathbf{P})
\end{aligned}$$

So

$$\begin{aligned}
\mathbf{M}(s \cdot \mathbf{x} \cdot \Delta_{\mathbf{B}}\mathbf{P} + (1-s) \cdot \mathbf{y} \cdot \Delta_{\mathbf{N}}\mathbf{P} + \mathbf{P}) = & \\
&\mathbf{M}(\mathbf{P}) - (1-s) \cdot \mathbf{y} \cdot \mathbf{M}(\mathbf{P}) - s \cdot \mathbf{x} \cdot \mathbf{M}(\mathbf{P}) \\
&+ s \cdot \mathbf{x} \cdot \mathbf{A} \cdot \Delta_{\mathbf{B}}\mathbf{P} \cdot \mathbf{P} \\
&+ \mathbf{A}(s \cdot \mathbf{x} \cdot \cdot \Delta_{\mathbf{B}}\mathbf{P} + (1-s) \cdot \mathbf{y} \cdot \Delta_{\mathbf{N}}\mathbf{P})^2 \\
&= (1 - \mathbf{y} + s(\mathbf{y} - \mathbf{x})) \cdot \mathbf{M}(\mathbf{P}) + s \cdot \mathbf{x} \cdot \mathbf{A} \cdot \Delta_{\mathbf{B}}\mathbf{P} \cdot \mathbf{P} \\
&\quad + \mathbf{A}(\mathbf{y} \cdot \Delta_{\mathbf{N}}\mathbf{P} + s \cdot (\mathbf{x} \cdot \Delta_{\mathbf{B}}\mathbf{P} - \mathbf{y}\Delta_{\mathbf{N}}\mathbf{P}))^2 \\
&= \underbrace{(1-y)\mathbf{M}(\mathbf{P}) + \mathbf{A}\tilde{\Delta}_{\mathbf{N}}\mathbf{P} \cdot \mathbf{P}^2}_{a}
\end{aligned}$$

$$+ s \cdot \underbrace{((\mathbf{y}-\mathbf{x})\cdot\mathbf{M}(\mathbf{P}) + \mathbf{A}\cdot\tilde{\Delta}_{\mathbf{B}}\mathbf{P}\cdot\mathbf{P} + \mathbf{A}\cdot\tilde{\Delta}_{\mathbf{N}}\mathbf{P}\cdot\tilde{\Delta}\mathbf{P} + \mathbf{A}\cdot\tilde{\Delta}\mathbf{P}\cdot\tilde{\Delta}_{\mathbf{N}}\mathbf{P})}_{\mathbf{b}}$$

$$+ s^2 \cdot \underbrace{\mathbf{A}\cdot(\tilde{\Delta}P)^2}_{\mathbf{c}}$$

$$= \mathbf{a} + s\cdot\mathbf{b} + s^2\mathbf{c}$$

Where $\tilde{\Delta}_{\mathbf{N}}\mathbf{P} = \mathbf{y}\cdot\Delta_{\mathbf{N}}\mathbf{P}$, $\tilde{\Delta}_{\mathbf{B}}\mathbf{P} = \mathbf{x}\Delta_{\mathbf{B}}\mathbf{P}$, $\tilde{\Delta}\mathbf{P} = \tilde{\Delta}_{\mathbf{B}}\mathbf{P} - \tilde{\Delta}_{\mathbf{N}}\mathbf{P}$. So

$$\begin{aligned}\|\mathbf{M}(\cdot)\|_F^2 &= \|\mathbf{a}\|_F^2 + s\left(\mathrm{tr}(\mathbf{a}^*\cdot\mathbf{b}) + \mathrm{tr}(\mathbf{b}^*\cdot\mathbf{a})\right) + s^2\cdot\left(\|\mathbf{b}\|_F^2 + \mathrm{tr}(\mathbf{a}^*\mathbf{c}) + \mathrm{tr}(\mathbf{c}^*\mathbf{a})\right)\\ &\quad + s^3\left(\mathrm{tr}(\mathbf{b}^*\cdot\mathbf{c}) + \mathrm{tr}(\mathbf{c}^*\cdot\mathbf{b})\right) + s^4\cdot\|\mathbf{c}\|_F^2\\ &= \|\mathbf{a}\|_F^2 + s\cdot 2\cdot\mathrm{tr}(\mathbf{a}^*\mathbf{b}) + s^2\cdot\left(\|\mathbf{b}\|_F^2 + 2\,\mathrm{tr}(\mathbf{a}^*\mathbf{c})\right) + s^3\cdot 2\,\mathrm{tr}(\mathbf{b}^*\cdot\mathbf{c}) + s^4\cdot\|\mathbf{c}\|_F^2\\ &\equiv \mathbf{t}(s)\end{aligned}$$

$$\mathbf{t}'(s) = 2\cdot\mathrm{tr}(\mathbf{a}^*\mathbf{b}) + 2\cdot(\|\mathbf{b}\|_F^2 + 2\cdot\mathrm{tr}(\mathbf{a}^*\mathbf{c})\cdot s + 6\cdot\mathrm{tr}(\mathbf{b}^*\cdot\mathbf{c})\cdot s^2 + 4\cdot\|\mathbf{c}\|_F^2\cdot s^3$$

Unsing the Dynare typology above

$$\mathbf{a} \quad {}_{n}= \overset{\begin{matrix} n^s & n^- & n^{++}\end{matrix}}{\begin{bmatrix} \mathbf{0} & \tilde{\mathbf{a}}_1 + \tilde{\mathbf{a}}_2 & \mathbf{0}\end{bmatrix}}$$

$$\underset{n\times n^-}{\tilde{\mathbf{a}}_1} = \begin{matrix} n^s \\ n^d\end{matrix}\overset{n^-}{\begin{bmatrix}\mathbf{0}\\ (1-\mathbf{y})\mathbf{x}_2\end{bmatrix}},\quad \underset{n^d\times n^-}{\mathbf{x}_2} = \underset{n^d\times n^+}{\tilde{\mathbf{A}}^+}\ \underset{n^+\times n}{+\mathbf{P}_{m/++,\bullet}}\ \underset{n\times n^-}{\mathbf{P}_{\bullet,--/m}} + \underset{n^d\times n^d}{\tilde{\mathbf{A}}^\circ}\ \underset{n^d\times n^-}{\mathbf{P}_{d,--/m}} + \underset{n^d\times n^-}{\tilde{\mathbf{A}}^-}$$

$$\underset{n\times n^-}{\tilde{\mathbf{a}}_2} = \mathbf{y}^2\cdot \begin{matrix} n^s \\ n^d\end{matrix}\overset{n^+}{\begin{bmatrix}\tilde{\mathbf{A}}^+\\ \tilde{\mathbf{A}}^+\end{bmatrix}}(\mathbf{d_N}\ \underset{n^+\times n}{\mathbf{P}_{m/++,\bullet}}\ \mathbf{d_N}\ \underset{n\times n^-}{\mathbf{P}_{\bullet,--/m}})$$

$$\mathbf{b} \quad {}_{n}= \overset{\begin{matrix} n^s & n^- & n^{++}\end{matrix}}{\begin{bmatrix} \mathbf{0} & \tilde{\mathbf{b}}_1 + \tilde{\mathbf{b}}_2 & \mathbf{0}\end{bmatrix}}$$

$$\underset{n\times n^-}{\tilde{\mathbf{b}}_1} = \begin{matrix} n^s \\ n^d\end{matrix}\overset{n^-}{\begin{bmatrix}\mathbf{0}\\ (y-x)x_2\end{bmatrix}}$$

$$\underset{n\times n^-}{\tilde{\mathbf{b}}_2} = \begin{matrix} n_s \\ n_d\end{matrix}\overset{n_+}{\begin{bmatrix}\check{A}^+\\ \tilde{A}^+\end{bmatrix}}\cdot\Big(\mathbf{xd_B}\ \underset{n^+\times n}{\mathbf{P}_{m/++,\bullet}}\cdot\ \underset{n\times n^-}{\mathbf{P}_{\bullet,--m}} + \mathbf{yd_N}\ \underset{n^+\times n}{\mathbf{P}_{m/++,\bullet}}\ (\mathbf{xd_B}\ \underset{n\times n^-}{\mathbf{P}_{\bullet,--/m}} - \mathbf{yd_N}\ \underset{n\times n^-}{\mathbf{P}_{\bullet,--/m}})$$

$$+\ \mathbf{y}\ (\mathbf{xd_B}\ \underset{n^+\times n}{\mathbf{P}_{m/++,\bullet}} - \mathbf{yd_N}\ \underset{n^+\times n}{\mathbf{P}_{m/++,\bullet}})\mathbf{d_N}\ \underset{n\times n^-}{\mathbf{P}_{\bullet,--/m}}\Big)$$

$$\mathbf{c} = n \begin{array}{c} \\ n^s \\ n^d \end{array} \overset{\begin{array}{ccc} n^s & n^+ & n^- \end{array}}{\left[\begin{array}{ccc} \mathbf{0} & \begin{array}{c} \check{\mathbf{A}}^+ \\ \tilde{\mathbf{A}}^+ \end{array} & (\mathbf{x}\, \mathbf{d_B}\, \underset{n^+ \times n}{\mathbf{P}_{m/++,\bullet}} - \mathbf{y}\, \mathbf{d_N}\, \underset{n^+ \times n}{\mathbf{P}_{m/++,\bullet}})\, (\mathbf{x}\, \mathbf{d_B}\, \underset{n^- \times n}{\mathbf{P}_{\bullet,--/m}} - \mathbf{y}\, d_{\mathbf{N}}\, \underset{n^- \times n}{\mathbf{P}_{\bullet,--/m}}) \end{array} \begin{array}{c} \\ \\ \mathbf{0} \end{array}\right]} \begin{array}{c} \\ \\ n_{++} \end{array}$$

6.4. Proof of Theorem 1

Following Higham (2008, Section 4.9.4.), local stability requires the Fréchet derivative of F at P to have bounded powers, which holds if its spectral radius is less than one. At P, (80) is

$$\operatorname{vec}(\mathscr{D}_P F(\Delta P)) = \left(\left[(AP+B)^{-1} C \right]' \otimes \left[(AP+B)^{-1} A \right] \right) \operatorname{vec}(\Delta P) \tag{A2}$$

$$= \left(P' \otimes \left[(AP+B)^{-1} A \right] \right) \operatorname{vec}(\Delta P) \tag{A3}$$

Hence the spectral radius is equal to the eigenvalue of $P' \otimes \left[(AP+B)^{-1} A \right]$ with the largest magnitude. As the eigenvalues of the Kronecker product of two matrices is the set of all crossproducts of the eigenvalues of the respective matrices, the spectral radius is the product of the eigenvalues of P and $(AP+B)^{-1} A$ with the largest magnitudes. Following Lan and Meyer-Gohde (2014, Corollary 4.2.), (19) can be factored as

$$M(\lambda) \equiv A\lambda^2 + B\lambda + C = (A\lambda + AP + B)(\lambda - P) \tag{A4}$$

Hence, the latent roots associated with $A\lambda + AP + B$ are those roots of $M(\lambda)$ not contained in set of eigenvalues of P. By assumption, the eigenvalues of P are inside the closed unit circle and the roots associated with $A\lambda + AP + B$ are outside the open unit circle. By inspection, the eigenvalues of $(AP+B)^{-1} A$ are the inverses of the roots associated with $A\lambda + AP + B$ and, therefore, are all inside the open unit circle. Hence the product of the largest magnitude eigenvalue of P and that of $(AP+B)^{-1} A$ is less than one in absolute value. That is, the spectral radius of the Fréchet derivative of F at P, the unique bounded solution, is less than one, completing the proof.

IMFS WORKING PAPER SERIES

Recent Issues

181 / 2023	Brian Fabo Martina Jančoková Elisabeth Kempf Luboš Pástor	Fifty Shades of QE: Robust Evidence
180 / 2023	Alexander Dück Fabio Verona	Robust frequency-based monetary policy rules
179 / 2023	Josefine Quast Maik Wolters	The Federal Reserve's Output Gap: The Unreliability of Real-Time Reliability Tests
178 / 2023	David Finck Peter Tillmann	The Macroeconomic Effects of Global Supply Chain Disruptions
177 / 2022	Gregor Boehl	Ensemble MCMC Sampling for Robust Bayesian Inference
176 / 2022	Michael D. Bauer Carolin Pflueger Adi Sunderam	Perceptions about Monetary Policy
175 / 2022	Alexander Meyer-Gohde Ekaterina Shabalina	Estimation and Forecasting Using Mixed-Frequency DSGE Models
174 / 2022	Alexander Meyer-Gohde Johanna Saecker	Solving linear DSGE models with Newton methods
173 /2022	Helmut Siekmann	Zur Verfassungsmäßigkeit der Veranschlagung Globaler Minderausgaben
172 / 2022	Helmut Siekmann	Inflation, price stability, and monetary policy – on the legality of inflation targeting by the Eurosystem
171 / 2022	Veronika Grimm Lukas Nöh Volker Wieland	Government bond rates and interest expenditures of large euro area member states: A scenario analysis
170 / 2022	Jens Weidmann	A new age of uncertainty? Implications for monetary policy
169 / 2022	Moritz Grebe Peter Tillmann	Household Expectations and Dissent Among Policymakers
168 / 2022	Lena Dräger Michael J. Lamla Damjan Pfajfar	How to Limit the Spillover from an Inflation Surge to Inflation Expectations?

167 / 2022	Gerhard Rösl Franz Seitz	On the Stabilizing Role of Cash for Societies
166 / 2022	Eva Berger Sylwia Bialek Niklas Garnadt Veronika Grimm Lars Other Leonard Salzmann Monika Schnitzer Achim Truger Volker Wieland	A potential sudden stop of energy imports from Russia: Effects on energy security and economic output in Germany and the EU
165 / 2022	Michael D. Bauer Eric T. Swansson	A Reassessment of Monetary Policy Surprises and High-Frequency Identification
164 / 2021	Thomas Jost Karl-Heinz Tödter	Reducing sovereign debt levels in the post-Covid Eurozone with a simple deficit rule
163 / 2021	Michael D. Bauer Mikhail Chernov	Interest Rate Skewness and Biased Beliefs
162 / 2021	Magnus Reif Mewael F. Tesfaselassie Maik Wolters	Technological Growth and Hours in the Long Run: Theory and Evidence
161 / 2021	Michael Haliassos Thomas Jansson Yigitcan Karabulut	Wealth Inequality: Opportunity or Unfairness?
160 / 2021	Natascha Hinterlang Josef Hollmayr	Classification of Monetary and Fiscal Dominance Regimes using Machine Learning Techniques
159 / 2021	Volker Wieland	The decline in euro area inflation and the choice of policy strategy
158 / 2021	Matthew Agarwala Matt Burke Patrycja Klusak Moritz Kraemer Kamiar Mohaddes	Rising Temperatures, Falling Ratings: The Effect of Climate Change on Sovereign Creditworthiness
157 / 2021	Yvan Lengwiler Athanasios Orphanides	Collateral Framework:Liquidity Premia and Multiple Equilibria
156 / 2021	Gregor Boehl Cars Hommes	Rational vs. Irrational Beliefs in a Complex World
155 / 2021	Michael D. Bauer Eric T. Swanson	The Fed's Response to Economic News Explains the "Fed Information Effect"
154 / 2021	Alexander Meyer-Gohde	On the Accuracy of Linear DSGE Solution Methods and the Consequences for Log-Normal Asset Pricing