A Broyden Rank $p+1$ Update Continuation Method with Subspace Iteration

T.L. van Noorden[†‡] S.M. Verduyn Lunel[§] A. Bliek[‡]

Abstract

In this paper we present an efficient branch following procedure that cannot only be used to compute branches of periodic solutions of periodically forced dynamical systems but also to determine the stability of the periodic solutions. The procedure combines Broyden's method in combination with a subspace iteration method to determine the dominant eigenvalues. The method has connections with the hybrid Newton-Picard methods developed by Lust et al. in [SIAM J. Sci. Comp. 19 (1998) pp. 1188–1209]. The method is applied to the computation of periodic states of a reverse flow reactor and its performance is compared with two variants of the hybrid Newton-Picard method.

Keywords: Broyden's method, periodically forced PDE's, reverse flow reactor, continuation of periodic solutions, subspace iteration.

1 Introduction

This paper is devoted to the efficient computation of periodic solutions of periodically forced partial differential equations and the determination of their stability. The partial differential equations of interest model cyclically operated chemical systems. Examples of such dynamical systems are pressure swing separation [14], the reverse flow reactor [7] and the pressure swing reactor [8].

In order to investigate the behaviour of dynamical systems numerically, we first have to discretize the equations. For a shooting method, first the spatial variable of the equations is discretized. This results in a large system of N (where N depends on the space discretization) time periodic ordinary differential equations. This system can abstractly be written as

$$\dot{x}(t) = f(t, x(t), \lambda), \;\; x(t) \in \mathbb{R}^N, \;\; \lambda \in \mathbb{R}, \quad \text{with } f(t, \cdot, \cdot) = f(t+1, \cdot, \cdot). \tag{1}$$

[†]Universiteit van Amsterdam, Department of Chemical Engineering, Nieuwe Achtergracht 166, 1018 WV Amsterdam, The Netherlands.

[‡]Vrije Universiteit Amsterdam, Department of Mathematics, De Boelelaan 1081A, 1081HV Amsterdam, The Netherlands.

[§]Universiteit Leiden, Department of Mathematics, Niels Bohrweg 1,2333 CA Leiden, The Netherlands. Corresponding author. email: verduyn@math.leidenuniv.nl

The right hand side, denoted by f, of the ordinary differential equation (1) has the following additional structure. If a finite element or finite difference discretization is used to discretize the spatial variable, then the components of x can be ordered in such a way that the Jacobian of f has a band structure. Secondly, the time variable t can always be rescaled in order to assume that the period of the periodic forcing in f is equal to one.

The map $F : \mathbb{R}^N \times \mathbb{R} \to \mathbb{R}^N$ that for a given λ assigns to the initial data at time zero, $x(0) = x_0$, the value of the solution after one period, i.e., the state $x(1)$, is called the Poincaré or period map of (1),

$$F(x_0, \lambda) = x(1, x_0), \tag{2}$$

where $x(t, x_0)$ denotes the solution of (1) with initial condition $x(0) = x_0$. So, evaluating the map F is equivalent to integrating the system (1) over one time unit starting with a given initial condition x_0.

A periodic state of the chemical system is a 1-periodic solution $x(t)$, i.e., a solution of (1) with $x(0) = x(1)$, and therefore the initial condition of a periodic solution of (1) is a solution of the fixed point equation

$$x = F(x, \lambda). \tag{3}$$

We define the Jacobian of F at the point $\boldsymbol{x} = (x, \lambda)$ to be the square matrix $J(x, \lambda)$ given by $J(x, \lambda) := \frac{\partial F}{\partial x}(x, \lambda)$. The stability of a solution x^* of (3) is determined by the eigenvalues of the Jacobian $J(x^*, \lambda)$ at the fixed point x^*.

If we look at solutions $\boldsymbol{x} = (x, \lambda)$ of the equation $F(x, \lambda) = x$, it is known that these solutions form in general, at least locally, a curve in $\mathbb{R}^{N+1}$, see for example [5]. Pseudo arc-length continuation [9] allows to compute this curve also past turning points. Suppose we have a solution $\boldsymbol{x}_0 = (x_0, \lambda_0)$ of $F(x, \lambda) = x$, as well as the normalized direction vector of the solution curve at (x_0, λ_0), which we will denote by $\dot{\boldsymbol{x}}_0 = (\dot{x}_0, \dot{\lambda}_0)$. Pseudo arc-length continuation consists of solving the following equations for x_1 and λ_1

$$\begin{cases} F(x_1, \lambda_1) - x_1 = 0, \\ (x_1 - x_0)^T \dot{x}_0 + (\lambda_1 - \lambda_0)\dot{\lambda}_0 - s = 0, \end{cases} \tag{4}$$

where s is the step size and $(x_1 - x_0)^T$ is the transpose of $(x_1 - x_0)$. It is not always possible to compute the direction vector $\dot{\boldsymbol{x}}_0 = (\dot{x}_0, \dot{\lambda}_0)$ directly, but a good approximation can be obtained from

$$\dot{\boldsymbol{x}}_0 \approx \frac{(\boldsymbol{x}_0 - \boldsymbol{x}_{-1})}{|(\boldsymbol{x}_0 - \boldsymbol{x}_{-1})|}, \tag{5}$$

where $\boldsymbol{x}_{-1}$ is a previously computed solution on the curve.

There exist various ways to obtain numerical solutions to the equations (3) or (4). When a periodic state $x(t, x^*)$ is stable, i.e., all the eigenvalues of the Jacobian $J(x^*, \lambda)$ are inside the open unit disc $D := \{z \in \mathbb{C} : |z| < 1\}$, then the periodic state can be obtained by integrating (1) on a long time interval, starting with an initial condition that is sufficiently close to the periodic solution. This approach is equivalent to a Picard iteration of the map F. This approach is, in many cases, not feasible, since the convergence to a periodic state of a dynamic simulation may be very slow [14]. Furthermore, only *stable* periodic states can be obtained by this approach.

A Newton-type method, applied directly to the fixed point equation (3), would allow to compute both stable and unstable periodic states. However, such an approach involves the computation or approximation of the Jacobian $J(x, \lambda)$ at various points x, either by numerical differentiation or by solving the variational form of (1). Both approaches are very expensive when N is large [10, 3]. Note that, in contrast to the map f in the right hand side of (1), the Jacobian $J(x, \lambda)$ does generally not have a band structure.

In numerical literature, a number of hybrid Newton-Picard methods have been developed, see [10, 6, 12]. These methods use the fact that for many physically interesting (infinite dimensional) systems, the interesting dynamics actually occurs only in a low dimensional subspace. A vector basis of this low dimensional subspace is obtained from a subspace iteration algorithm. This approach also yields eigenvalues that determine the stability. Furthermore, the action of the Jacobian $J(x, \lambda)$ is only computed on the space spanned by this basis. It turns out that the theoretical convergence rate of the Newton-Picard method is linear.

In the chemical engineering literature other methods have been proposed to solve the equations (3) or (4) more efficiently. It has been shown that Broyden's method is very efficient in most cases [13]. Broyden's method uses only one evaluation of F per iteration and converges superlinearly. However, the disadvantage of Broyden's method is that the approximations, that the method produces, to the Jacobian $J(x^*, \lambda)$ of F at the periodic solution (x^*, λ) are not accurate enough for the determination of the stability of the periodic solution.

In this paper we discuss a Broyden rank $p+1$ update continuation method with subspace iteration (in short the BSI rank $p+1$ method). The method combines the efficiency of Broyden's method with the ability of the subspace iteration method to determine the stability of the computed solutions. The application of both the BSI rank $p+1$ method and the Newton-Picard method to an example system, shows that the BSI rank $p+1$ method is approximately twice as efficient as the Newton-Picard method, in terms of number of evaluations of F. The difference in efficiency lies mainly in the fact that the BSI method uses less evaluations of F per iteration than the Newton-Picard method. The example shows that the convergence rate and robustness of the two methods are comparable.

Since the value $F(x_0, \lambda)$ is obtained by integrating a large system of differential equations over a period of time with initial condition x_0, the evaluation of F is a computationally expensive task. This means that for any method used to find a fixed point of F, most of the computation time will be spent on the evaluation of F. Thus the method that needs the least number of evaluations of F will be the most efficient method. Therefore the comparison between the BSI rank $p+1$ method is made with respect to the number of evaluations of F needed to compute a branch of periodic solutions.

The organization of this paper is as follows. In Section 2 we introduce the BSI rank $p+1$ method and discuss the merits of the method in comparison with the Newton-Picard method. In Section 3 we present the model equations for the cooled reverse flow reactor. In Section 4 we compute a branch of periodic states of the reverse flow reactor using both the BSI rank $p+1$ method and the Newton-Picard method, and compare the efficiency of both methods.

2 BSI Rank $p+1$ Method

In the Introduction we saw that, for the computation of a branch of periodic solutions of (1), we have to solve repetitively systems of the form

$$\begin{pmatrix} F(x,\lambda)-x \\ w^T(x-y)+\kappa(\lambda-\mu)-s \end{pmatrix} = 0, \tag{6}$$

where $w,\ y \in \mathbb{R}^N$ and $\kappa,\ \mu,\ s \in \mathbb{R}$ are given. For convenience, we define the map $G:\mathbb{R}^{N+1}\to\mathbb{R}^{N+1}$ by

$$G(\boldsymbol{x}) = G(x,\lambda) := \begin{pmatrix} F(x,\lambda)-x \\ w^T(x-y)+\kappa(\lambda-\mu)-s \end{pmatrix}, \tag{7}$$

where $\boldsymbol{x}$ is the $N+1$ dimensional vector (x,λ).

An efficient way to compute solutions to the system $G(\boldsymbol{x}) = 0$ is to use Broyden's "good" method [1] (there also exist other variants of Broyden's method, e.g. see [4]). Broyden's method produces approximations to a zero of G using the following iteration scheme

$$\boldsymbol{x}_{i+1} = \boldsymbol{x}_i + H_i G(\boldsymbol{x}_i)\ , \tag{8}$$

with $H_i \in \mathbb{R}^{(N+1)\times(N+1)}$ iterative approximations to $-\frac{\partial G}{\partial \boldsymbol{x}}(\boldsymbol{x}_i)^{-1}$ defined by

$$H_{i+1} = H_i - \frac{(\boldsymbol{p}_i + H_i\boldsymbol{g}_i)\boldsymbol{p}_i^T H_i}{\boldsymbol{p}_i^T H_i \boldsymbol{g}_i}\ , \tag{9}$$

where $\boldsymbol{g}_i = G(\boldsymbol{x}_{i+1}) - G(\boldsymbol{x}_i)$ and $\boldsymbol{p}_i = \boldsymbol{x}_{i+1} - \boldsymbol{x}_i$. This iteration scheme for H_i is derived from the following constraints on the update H_{i+1} of H_i:

$$-H_{i+1}^{-1}\boldsymbol{p}_i = \boldsymbol{g}_i, \tag{10}$$

$$-H_{i+1}^{-1}\boldsymbol{y} = -H_i^{-1}\boldsymbol{y} \quad \text{for all } \boldsymbol{y} \perp \boldsymbol{p}_i. \tag{11}$$

Equation (10) can be viewed as a secant approximation of $\frac{\partial G}{\partial \boldsymbol{x}}(\boldsymbol{x}_{i+1})$ in the direction $\boldsymbol{p}_i$. Note that the only information that is used in updating the approximation H_i at the new approximation $\boldsymbol{x}_{i+1}$ is the function value $G(\boldsymbol{x}_{i+1})$. The method thus uses only one evaluation of F in each iteration. However, there is no guarantee that, if the approximations $\boldsymbol{x}_i$ converge to a solution $\boldsymbol{x}^*$ of $G(\boldsymbol{x}) = 0$, the approximations H_i converge to $-\frac{\partial G}{\partial \boldsymbol{x}}(\boldsymbol{x}^*)^{-1}$. This means that the approximations H_i cannot be used to obtain good approximations of the eigenvalues of $\frac{\partial F}{\partial x}(\boldsymbol{x}^*)$, which are needed for the determination of the stability of the solution $\boldsymbol{x}^*$. In this section we discuss an approach to combine Broyden's method with a subspace iteration algorithm [11] so that also approximations to the largest eigenvalues of $\frac{\partial F}{\partial x}(\boldsymbol{x}^*)$ can be computed accurately.

For this approach we do not only need the initial approximations $\boldsymbol{x}_0$ and H_0, required for Broyden's method, but also initial approximations to the p largest eigenvalues of $\frac{\partial F}{\partial x}(\boldsymbol{x}_0)$ together with an initial approximation $V_0 \in \mathbb{R}^{N\times p}$ for the orthonormal basis of the subspace spanned by the p corresponding eigenvectors. In the first step of our new approach, we proceed as in Broyden's method, and compute the next approximation to the solution of (6) with the same formula as before

$$\boldsymbol{x}_1 \quad = \quad \boldsymbol{x}_0 + H_0 G(\boldsymbol{x}_0). \tag{12}$$

Now we would like to update H_0, V_0 and the approximations of the eigenvalues.

Let us first consider the updating of the eigenvalues and of V_0. We would like the updates of the eigenvalues to approximate the p largest eigenvalues of the Jacobian $\frac{\partial F}{\partial x}(\boldsymbol{x}_1)$ in the new approximation $\boldsymbol{x}_1$ and the update V_1 to approximate the basis of the subspace spanned by the corresponding eigenvectors. Both the updates of the eigenvalues and of V_0 can be obtained by performing one (or more) subspace iteration(s). The version of the subspace iteration algorithm that we use, is the version also used in [10]. The algorithm can be summarized by the following steps.

Subspace iteration with projection

1) Set $\tilde{V}_0 = V_0$ and set $j = 0$

2) Compute $W_j = \frac{\partial F}{\partial x}(\boldsymbol{x}_1)\tilde{V}_j$.

3) Compute $S_j = \tilde{V}_j^T W_j$.

4) Compute the Schur vectors $Y_j = [y_1, ..., y_p]$ of S_j, order them according to decreasing modulus of the corresponding eigenvalue.

5) Compute $\tilde{V}_{j+1} = W_j Y_j$.

6) Orthonormalize $\tilde{V}_{j+1}$.

7) If $\tilde{V}_{j+1}$ is accurate enough, then stop and return $\tilde{V}_{j+1}$ and W_j. Else set $j = j + 1$ and go to step 2).

In this procedure the matrices $\tilde{V}_j$ will converge to a orthonormal basis of the subspace spanned by the eigenvectors and generalized eigenvectors associated with the p eigenvalues of $\frac{\partial F}{\partial x}(\boldsymbol{x}_1)$ with the largest modulus. The convergence properties of the subspace iteration algorithm are discussed in [11]. In each iteration of the BSI rank $p+1$ method, we only perform one iteration of the subspace iteration algorithm. This iteration provides us with new approximations to the largest eigenvalues of $\frac{\partial F}{\partial x}(\boldsymbol{x}_1)$, with a new orthonormal basis $V_1 = \tilde{V}_1$ and with the matrix

$$W_0 = \frac{\partial F}{\partial x}(\boldsymbol{x}_1)\tilde{V}_0 = \frac{\partial F}{\partial x}(\boldsymbol{x}_1)V_0. \tag{13}$$

This last matrix is very useful in updating H_0. If we define the matrices

$$\boldsymbol{V} := \begin{pmatrix} V_0 \\ 0 \end{pmatrix} \text{ and } \boldsymbol{Z} := \begin{pmatrix} W_0 - V_0 \\ w^T V_0 \end{pmatrix}, \tag{14}$$

then it is easily seen that

$$\frac{\partial G}{\partial \boldsymbol{x}}(\boldsymbol{x}_1)\boldsymbol{V} = \boldsymbol{Z}. \tag{15}$$

We can use the equation (15) in the same way as the constraint (10) is used in Broyden's method, to update H_0. So, analogously to the Broyden rank one update, we would now like to update H_0 in such a way that

$$-H_1^{-1}\boldsymbol{V} = \boldsymbol{Z}, \tag{16}$$

$$-H_1^{-1}(\boldsymbol{x}_1 - \boldsymbol{x}_0) = \big(G(\boldsymbol{x}_1) - G(\boldsymbol{x}_0)\big), \tag{17}$$

$$H_1^{-1}\boldsymbol{q} = H_0^{-1}\boldsymbol{q} \text{ for all } \boldsymbol{q} \text{ with } \boldsymbol{V}^T\boldsymbol{q} = 0 \text{ and } (\boldsymbol{x}_1 - \boldsymbol{x}_0)^T\boldsymbol{q} = 0. \tag{18}$$

As long as H_0 is invertible, $(I - \boldsymbol{V}\boldsymbol{V}^T)(\boldsymbol{x}_1 - \boldsymbol{x}_0) \neq 0$ and $(I - \boldsymbol{Z}\boldsymbol{Z}^T)\big(G(\boldsymbol{x}_1) - G(\boldsymbol{x}_0)\big) \neq 0$, the matrix H_1^{-1} characterized by the equations (16)-(18) is well defined and unique. If H_1^{-1}, defined by (16)-(18), is invertible, then its inverse H_1 can be computed as follows. First we define $B_0 := H_0$ and then recursively

$$B_{l+1} \;:=\; B_l - \frac{(\boldsymbol{v}_l + B_l\boldsymbol{z}_l)\boldsymbol{v}_l^T B_l}{\boldsymbol{v}_l^T B_l \boldsymbol{z}_l}, \tag{19}$$

up to B_p, where $\boldsymbol{v}_l$ and $\boldsymbol{z}_l$ denote the lth column of respectively $\boldsymbol{V}$ and $\boldsymbol{Z}$. Now, the update H_1 is given by

$$H_1 \;=\; B_p - \frac{(\boldsymbol{p} + B_p\boldsymbol{f})\boldsymbol{p}^T B_p}{\boldsymbol{p}^T B_p \boldsymbol{f}}, \tag{20}$$

where we use the notation

$$\boldsymbol{p} \;=\; (I - \boldsymbol{V}\boldsymbol{V}^T)(\boldsymbol{x}_1 - \boldsymbol{x}_0), \tag{21}$$

$$\boldsymbol{f} \;=\; G(\boldsymbol{x}_1) - G(\boldsymbol{x}_0) - \boldsymbol{Z}\boldsymbol{V}^T(\boldsymbol{x}_1 - \boldsymbol{x}_0). \tag{22}$$

At this point we have computed the new updates $\boldsymbol{x}_1$, V_1, H_1 and updates of the p largest eigenvalues. By following the above procedure starting with these updates, we can again obtain another set of updates, and so on. This iteration of the above described procedure defines the BSI rank $p+1$ method, and can be summarized by the following steps:

BSI rank $p+1$ method

1) Supply $H_0 \in \mathbb{R}^{(N+1)\times(N+1)}$, invertible; $V_0 \in \mathbb{R}^{N\times p}$, orthonormal; $\boldsymbol{x}_0 = (x_0, \lambda_0) \in \mathbb{R}^{(N+1)}$, and set $i = 0$.

2) Compute

$$\boldsymbol{x}_{i+1} = \boldsymbol{x}_i + H_i G(\boldsymbol{x}_i).$$

(costs one evaluation of F)

3) Subspace iteration: compute $S := \frac{\partial F}{\partial x}(\boldsymbol{x}_{i+1})V_i$ (costs p evaluations of F). Compute the ordered Schur-factorization $Y^T RY$ of VS. Compute $V = SY$. Orthonormalize the columns of V and put the result in V_{i+1}.

4) Compute

$$H_{i+1} = \begin{cases} B_p - \dfrac{(\boldsymbol{p} + B_p \boldsymbol{f})\boldsymbol{p}^T B_p}{\boldsymbol{p}^T B_p \boldsymbol{f}} & \text{if } \boldsymbol{p}^T B_p \boldsymbol{f} \neq 0 \\ & \text{and } (I - \boldsymbol{Z}\boldsymbol{Z}^T)\big(G(\boldsymbol{x}_{i+1}) - G(\boldsymbol{x}_i)\big) \neq 0, \\ B_p & \text{otherwise,} \end{cases}$$

where B_p is defined recursively by $B_0 = H_i$ and

$$B_{l+1} = \begin{cases} B_l - \dfrac{(\boldsymbol{v}_l + B_l \boldsymbol{z}_l)\boldsymbol{v}_l^T B_l}{\boldsymbol{v}_l^T B_l \boldsymbol{z}_l} & \text{if } \boldsymbol{v}_l^T B_l \boldsymbol{z}_l \neq 0, \\ B_l & \text{if } \boldsymbol{v}_l^T B_l \boldsymbol{z}_l = 0, \end{cases}$$

with $\boldsymbol{v}_l$ and $\boldsymbol{z}_l$ the lth column of respectively

$$\boldsymbol{V} := \begin{pmatrix} V_i \\ 0 \end{pmatrix} \text{ and } \boldsymbol{Z} := \begin{pmatrix} \big(\frac{\partial F}{\partial x}(\boldsymbol{x}_{i+1}) - I\big)V_i \\ w^T V_i \end{pmatrix}.$$

and where

$$\begin{aligned} \boldsymbol{p} &= (I - \boldsymbol{V}\boldsymbol{V}^T)(\boldsymbol{x}_{i+1} - \boldsymbol{x}_i), \\ \boldsymbol{f} &= G(\boldsymbol{x}_{i+1}) - G(\boldsymbol{x}_i) - \boldsymbol{Z}\boldsymbol{V}^T(\boldsymbol{x}_{i+1} - \boldsymbol{x}_i). \end{aligned}$$

Set $i = i + 1$, go to 2).

This algorithm needs $p+1$ evaluations of F per iteration. In Section 4 we will use the algorithm to compute a branch of periodic states of a reverse flow reactor.

In order to give an idea of the efficiency of the BSI rank $p+1$ method, we compare its performance with the performance of two variants of a Newton-Picard method developed in [10]. This method uses the fact that for many physically interesting (infinite dimensional) systems, the interesting dynamics actually occurs only in a low dimensional subspace. Numerical evidence that this is also a feature for many periodic chemical processes is presented in [15]. The first variant of the method is called CNP in [10]. This variant needs $p + 2l + 4$ evaluations of F per iteration, where l is a parameter of the method that specifies the number of Picard iterations used in one full iteration of the Newton-Picard method. The second variant is called CNPGS. This variant needs $p+l+2$

Table 1: The dimensionless parameter values for the reverse flow reactor

parameter	value
K_1	$6.9393 \cdot 10^{-4}$
K_2	0.1749
K_3	$1.5577 \cdot 10^{-6}$
K_4	variable
K_5	$2.4038 \cdot 10^{-3}$
K_6	174.06
K_7	0.01
$g(\theta)$	$\dfrac{1.6656 \cdot 10^{-5} e^{25.785(\theta-1)/\theta}}{1.6656 \cdot 10^{-5} + e^{-25.785/\theta}}$

evaluations of F per iteration and so is less expensive than CNP. In [10] it is reported that the CNP and CNPGS have the same asymptotic convergence rate, but that CNP is the more robust method (i.e., has a larger domain of attraction) and thus allows larger step sizes in the continuation algorithm.

In [10], the subspace iteration is performed on $p + p_e$ (with $p_e = 2$, 3 or 4) vectors in order to accelerate the convergence of the subspace iteration procedure. In [10], also the number p is varied during the computations in order to ensure that the subspace iteration is only performed on the subspace spanned by eigenvectors of which the corresponding eigenvalues are outside the region $D_\rho := \{z \in \mathbb{C} : |z| < \rho\}$ (usually ρ is taken to be 0.5). Here, for both the Newton-Picard methods and the BSI method, we only perform the subspace iteration on p vectors, and keep this number fixed during the computations. This approach works nicely for the cooled reverse flow reactor (see [15]). However, for dynamical systems with more wildly behaving eigenvalues, a procedure for varying p as used in [12, 6] or in [10] is necessary.

In [10] it is reported that the Newton-Picard method converges linearly. Broyden's method has theoretically better convergence properties. In [2] it is shown that Broyden's method converges locally at least Q-superlinearly, and in [4] it is shown that the method enjoys local $2N$-step, Q-quadratic convergence.

Another difference between our approach and that of [10] is that we do not use locking and deflation in the subspace iteration. Also the matrix-vector products $\frac{\partial F}{\partial x}(\boldsymbol{x}_{i+1})V_i$ in step 3) in the algorithm above, are calculated using a finite difference approximation given by

$$\frac{\partial F}{\partial x}(\boldsymbol{x})v \approx \epsilon^{-1}\big(F(x + \epsilon v, \lambda) - F(x, \lambda)\big),$$

instead of solving a variational form of (1).

3 The Cooled Reverse Flow Reactor

A Reverse Flow Reactor (RFR) is a packed bed reactor in which the flow direction is periodically reversed to trap a hot zone within the reactor. In this way exothermic reactions can be operated without preheating the feed stream. We consider a cooled RFR in which a single irreversible, exothermic, first order reaction occurs. We describe the RFR by a one-dimensional, pseudo-homogeneous model that accounts for axial heat and mass dispersion and external mass transfer resistance between the fluid and the catalyst. The basic model assumption is to assume that all the physical properties are independent of the temperature and the concentration. The model is taken from [7].

The variables in the model are the dimensionless temperature $\theta(t,x) : [0,\infty) \times [0,1] \to [0,\infty)$ and the conversion $\chi(t,x) : [0,\infty) \times [0,1] \to [0,1]$. The dimensionless parameters that appear in the model equations are denoted by K_j for $j = 1,..,7$, and by $g(\theta(t,x))$. The model specific values are given in Table 1. The parameter K_4 is used as the bifurcation parameter. For flow from left to right the dimensionless energy and species balances read

$$\theta_t = K_1\theta_{xx} - K_2\theta_x + K_3 g(\theta)(1-\chi) + K_4(1-\theta), \quad (23)$$

$$\chi_t = K_5\chi_{xx} - K_6\chi_x + K_7 g(\theta)(1-\chi). \quad (24)$$

with boundary conditions

$$\begin{array}{cc} K_1\theta_x(t,0) = K_2(\theta(t,0)-1), & K_5\chi_x(t,0) = K_6\chi(t,0), \\ \theta_x(t,1) = 0, & \chi_x(t,1) = 0. \end{array} \quad (25)$$

At each integer value of t the flow direction reverses and the evolution equations and the boundary conditions change accordingly. A *cycle* consists of two flow reversals. This means that one cycle has a duration of two time units. The equations (23)-(24) with the boundary conditions (25) describe the model.

A periodic state of (23)-(24) with the boundary conditions (25) is a solution that satisfies

$$\theta(0,x) = \theta(1,1-x) \quad \text{and} \quad \chi(0,x) = \chi(1,1-x). \quad (26)$$

Thus a periodic solution consist of two symmetric parts of one time unit.

4 Results and Comparison of the Methods

For the reverse flow reactor we chose to discretize the spatial variable on a grid of 60 nodes using a finite volume approach with upwinding of the first space derivatives in the mass balance. The discretization of the model equations results for the RFR in a system of 120 ordinary differential equations. Hence $N = 120$. The system of ODE's is integrated in time using the NAG fortran library routine D02EJF. In this section we present the results for the different methods used to compute a branch of periodic states of the reverse flow reactor. The branch is depicted in Figure 1. Here stable periodic states are represented by solid lines and unstable states by a dashed line. The different bifurcation points are denoted in the figure and explained in the caption.

We used three different values of p, i.e. 5, 7 and 9, for all the methods. For the CNP and the CNPGS we also varied the parameter l.

For all methods we used the same simple step size control procedure. After a convergence failure, the step size is decreased and divided by two and when a point is computed

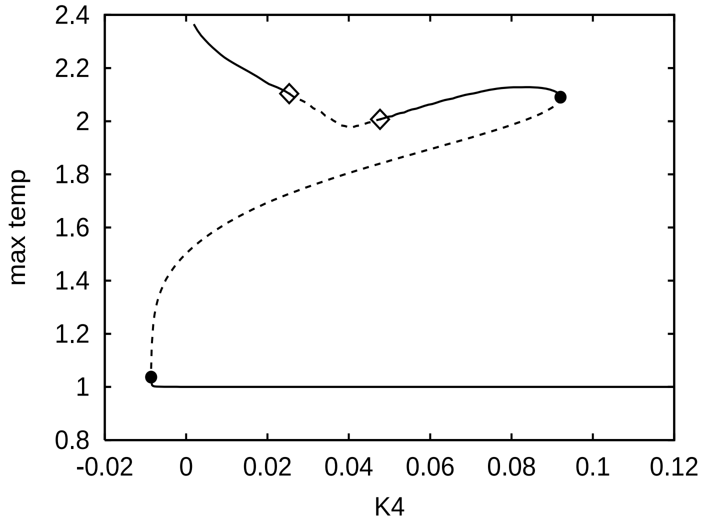

Figure 1: Branch of periodic solutions for the reverse flow reactor. Along the x-axis the parameter K_4 is plotted and along the y-axis the maximal dimensionless temperature in the bed at flow reversal is plotted. solid line: stable periodic states, dashed line: unstable states. $\diamond$: an eigenvalue crosses the unit circle at -1, $\bullet$: an eigenvalue crosses the unit circle at 1.

sufficiently fast, the step size is increased and multiplied by 1.6 (up to a maximum step size). This variable step size strategy allows us to observe not only differences in convergence speed, but also differences in the size of the domain of attraction of the different methods. For each point (x, λ) computed on the branch, the three largest eigenvalues of $J(x, \lambda)$ (according to absolute value) are computed with four digits of accuracy. For this purpose after the BSI or Newton-Picard iterations in most cases only one subspace iteration was sufficient.

The number of evaluations of F that each method needs to compute the whole branch is listed in Table 2. The number of points computed on the curve are listed in Table 3. We clearly see that the BSI rank $p+1$ method is the most efficient method. Actually, the BSI rank $p+1$ method is most efficient for $p = 5$, but also for the other values of p the method is much more efficient than the either the CNP or the CNPGS method.

The next most efficient method is CNP with $l = 2$ and $p = 7$. Note that there is not much difference between the $p = 7$ and $p = 9$ cases. For $p = 5$, however, the CNP method needs considerably more function evaluations.

The CNPGS method is the least efficient method and needs the most function evaluations. This is clearly due to the fact that it has the smallest domain of attraction. As is seen from Table 3, the CNPGS method needs considerably more continuation points than both the BSI and the CNP method. This fact is also illustrated in Figure 2. From this figure, we observe that the BSI rank $p+1$ method uses the maximum step size all along the branch except close to the turning points. The CNP method is less robust, as the method uses smaller step sizes, than the BSI rank $p+1$ method in a neighbourhood of the left turning point. The CNPGS method is the least robust method, since it is the only method that uses smaller step sizes to compute the unstable part of the branch between

Table 2: Number of evaluations of F for the different methods

method	# F eval. for $p = 5$	# F eval. for $p = 7$	# F eval. for $p = 9$
BSI	11551	12228	13557
CNP (l=1)	40296	22268	20900
CNP (l=2)	32551	19992	20744
CNP (l=3)	26504	20832	21964
CNPGS (l=1)	30784	28428	29032
CNPGS (l=2)	31292	30266	29623
CNPGS (l=3)	27797	28091	31550

Table 3: Number of continuation points for the different methods

method	# points for $p = 5$	# points for $p = 7$	# points for $p = 9$
BSI	342	341	340
CNP (l=1)	464	355	359
CNP (l=2)	420	345	359
CNP (l=3)	356	350	360
CNPGS (l=1)	415	411	396
CNPGS (l=2)	419	438	408
CNPGS (l=3)	403	395	409

the two points where an eigenvalues crosses the unit circle at -1.

In [10] it is noted that the performance of the Newton-Picard methods in terms of the number of evaluations of F needed, is essentially independent of the number of nodes used in the discretization. In the order to check whether this is also the case for the BSI rank $p+1$ method, we computed the same branch of solutions using a discretization of model equations on 100 nodes, so that $N = 200$. In this case the BSI rank $p+1$ method with $p = 7$ needed 12714 evaluations of F to compute the whole branch. This number is close to the number of evaluations of F needed for the discretization on 60 nodes.

5 Conclusions

In this paper we have introduced a Broyden rank $p+1$ update continuation method with subspace iteration (in short the BSI rank $p+1$ method) for the computation of branches of periodic solutions of periodically forced PDE's. The method makes efficiently use of the ideas behind Broyden's method and behind the Newton-Picard methods developed by [10]. As the name suggests, the method combines a Broyden method with a subspace iteration procedure for the computation of invariant subspaces. Simultaneously approximations to a periodic solution and to the eigenvalues that determine the stability are computed. In this way bifurcation points can be detected accurately and efficiently.

We have used the method to compute a branch of periodic states of a cooled reverse flow reactor (see Fig. 1). We have also used two variants of the Newton-Picard method [10] to compute the same branch. For the example system, the BSI rank $p+1$ method proved to be the most efficient method, in terms of number of period map evaluations.

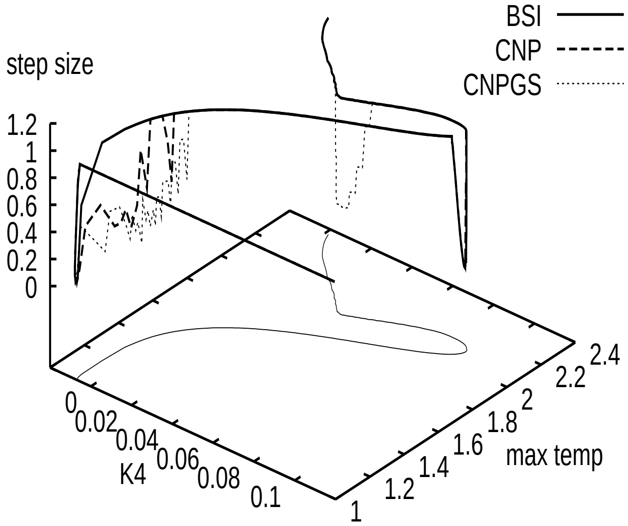

Figure 2: The step sizes for the different methods along the branch. Along the x-axis the parameter K_4 is plotted, along the y-axis the maximal dimensionless temperature in the bed at flow reversal is plotted and along the z-axis the step size used to compute the point on the branch is plotted. The maximum step size is equal to one. For each method the most efficient run is used; BSI: $p = 7$, CNP: $l = 2$ and $p = 7$, CNPGS: $l = 3$ and $p = 5$.

In fact, the BSI method needed approximately half the number of evaluations of F as the next most efficient method (CNP with $l = 2$ and $p = 7$). The main advantage of the BSI rank $p + 1$ method, when compared to the Newton-Picard methods, is the small number of period map evaluations per iteration of the method. The method proved also to have the largest domain of attraction. This results in the largest average step size, so that the method needed the fewest continuation points.

Of the two variants of the Newton-Picard method, the CNP and the CNPGS variant, the CNPGS is the least expensive variant in terms of number of evaluations of F per iteration. The CNP method, however, is also the least robust method of the two, and this results in the poorest overall performance.

We showed that the performance of the BSI rank $p + 1$ method in terms of number of evaluations of F is essentially independent of the number of nodes used in the discretization. This fact is already known for the Newton-Picard methods, see [10].

We believe that a more sophisticated variant of the subspace iteration with locking and deflation and an adaptive control of the dimension of the subspace on which the subspace iteration is performed, such as used in the methods of [10], can also be applied in the BSI rank $p + 1$ method. This will increase the efficiency of the new method even further.

References

[1] C. G. Broyden, *A class of methods for solving nonlinear simultaneous equations*, Mathematics of Computation, 19 (1965), pp. 577–593.

[2] C. G. Broyden, J. E. Dennis, and J. J. Moré, *On the local and superlinear convergence of quasi-Newton methods*, J. Inst. Math. Appl., 12 (1973), pp. 223–245.

[3] D. T. Croft and M. G. Levan, *Periodic states of adsorption cycles–I. Direct determination and stability*, Chemical Engineering Science, 49 (1994), pp. 1821–1829.

[4] D. M. Gay, *Some convergence properties of Broyden's method*, SIAM Journal on Numerical Analysis, 16 (1979), pp. 623–630.

[5] K. Georg, *On tracing an implicitely defined cirve by quasi-newton steps and calculating bifurcation by local perturbations*, SIAM J. Sci. Stat. Comput., 2 (1981), pp. 35–50.

[6] H. Jarausch and W. Mackens, *Solving large nonlinear systems of equations by an adaptive condensation process*, Numerische Mathematik, 50 (1987), pp. 633–653.

[7] J. Khinast, A. Gurumoorthy, and D. Luss, *Complex dynamic features of a cooled reverse-flow reactor*, AIChE Journal, 44 (1998), pp. 1128–1140.

[8] A. J. Kodde and A. Bliek, *Selectivity enhancement in consecutive reactions using the pressure swing reactor*, Stud. Surf. Sci. Catal., 109 (1997), pp. 419–428.

[9] Y. A. Kuznetsov, *Elements of applied bifurcation theory*, vol. 112 of Appl. Math. Sciences, Springer-Verlag New York, Inc., 1995.

[10] K. Lust, D. Roose, A. Spence, and A. R. Champneys, *An adaptive Newton-Picard algorithm with subspace iteration for computing periodic solutions*, SIAM Journal on Scientific Computing, 19 (1998), pp. 1188–1209.

[11] Y. Saad, *Numerical Methods for Large Eigenvalue Problems*, Algorithms and Architectures for Advanced Scientific Computing, Manchester University Press, Manchester, 1992.

[12] G. Shroff and H. Keller, *Stabilization of unstable procedures: The recursive projection method*, SIAM Journal on Numerical Analysis, 30 (1993), pp. 1099–1120.

[13] T. L. Van Noorden, S. M. Verduyn Lunel, and A. Bliek, *Direct determination of cyclic steady states of cyclically operated packed bed reactors*, in Volume 1: Molecular Properties, Reaction Diffusion Problems, Reaction Engineering, Simulation of Reactive Flows, F. Keil, W. Mackens, H. Voss, and J. Werther, eds., Springer-Verlag, 1999, pp. 311–318.

[14] ——, *Acceleration of the determination of periodic states of cyclically operated reactors and separators*, Tech. Report WS-549, Vrije Universiteit, Amsterdam, 2000.

[15] ——, *The efficient computation of periodic states of cyclically operated chemical processes*, Tech. Report MI 2001-22, Mathematical Institute, University of Leiden, 2001.