Control Theoretic Techniques for Stepsize Selection in Explicit Runge–Kutta Methods

KJELL GUSTAFSSON
Lund Institute of Technology

The problem of stepsize selection in the numerical solution of ordinary differential equations can be viewed as an automatic control problem. We will demonstrate how control theory can be used to analyze and improve the standard stepsize control algorithm.

Previously, Gustafsson et al. [5] suggested a PI controller to overcome the problem of oscillating stepsize sequences that typically appear when explicit Runge-Kutta methods encounter stiffness. Its properties were investigated experimentally. Here, the superior properties of the PI controller will be analyzed using a model for the relation between stepsize and the local truncation error in the integration method. When stability limits the stepsize, the standard asymptotic model fails to correctly describe this relation. Instead, a dynamic model that takes this behavior into account is derived for explicit Runge-Kutta methods. The model is verified using numerical tests and system identification.

The derived model helps analyzing standard stepsize control. The analysis gives insight and leads to a good understanding of the properties of the control system. The acquired understanding is used to further improve the PI controller as well as tuning its parameters.

Categories and Subject Descriptors: G.1.7 [**Numerical Analysis**]: Ordinary Differential Equations—*initial value problems, single step methods*; G.4 [**Mathematics of Computing**]: Mathematical Software—*algorithm analysis, efficiency, reliability and robustness*

General Terms: Algorithms, Theory

Additional Key Words and Phrases: Control theory, numerical integration, Runge-Kutta methods, stability, stepsize control, stiff systems

1. INTRODUCTION

A standard algorithm for stepsize control is [4, 7]

$$h_n = \gamma \left(\frac{tol}{r_n} \right)^{1/k} h_{n-1} \tag{1}$$

This work was supported by the Swedish National Board for Technical Development (STU) under contract 88-01611P. Author's address: Department of Automatic Control, Lund Institute of Technology, P.O. Box 118, S-221 00 Lund, Sweden.

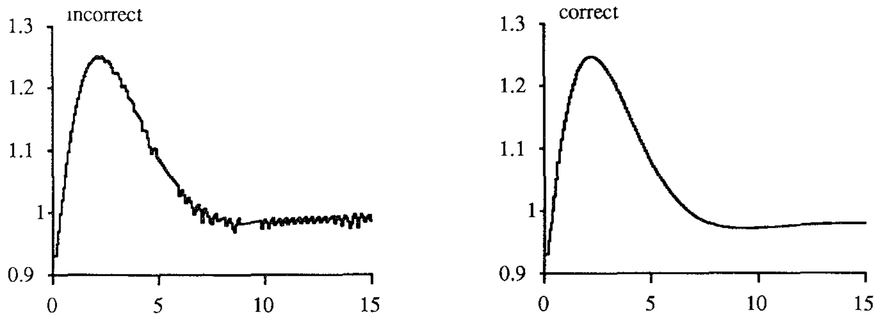

Fig. 1. A signal drawn from a control system simulation. The oscillatory component, in the signal to the left, is caused by an irregular stepsize sequence. The correct signal, to the right, is obtained by improving the stepsize control algorithm

where h is the stepsize, r the estimated error, *tol* the user-specified tolerance, γ a safety factor reducing the risk of a rejected step and k related to the order of the integration method. This algorithm normally performs quite well, but there are exceptions. One such exception is when stability rather than accuracy limits the stepsize. This often results in a stepsize sequence that oscillates violently. Moreover, the nonsmooth stepsize sequence may excite modes that the error estimator fails to recover properly, leading to an erroneous solution. Figure 1 shows an example from a simulation of a small control system (see Section 6). By improving the stepsize control algorithm the artifact can be removed, yielding a correct simulation result.

Another case, for which the standard stepsize controller does not perform well, is differential equations with drastic changes in behavior. Here, one often finds long sequences of alternating accepted and rejected steps, resulting in wasted computing time. A simulation of the Brusselator (see Section 5) is shown in Figure 2. Just before and during the large state transitions at $t = 24.5$, there are many rejected steps, which partly can be attributed to poor stepsize control.

Earlier studies of the problem of oscillating stepsize sequences have focused mainly on describing, characterizing and analyzing the behavior [8, 9, 10, 13]. An exception is the recent study [11], in which an explicit Runge–Kutta method is modified to behave properly together with the standard stepsize controller. The modification trades the stability region and/or the precision of the integration method for a more well-behaved stepsize sequence. Other interesting early studies are presented by Watts [15] and Zonneveld [16] where an extrapolation scheme on stepsizes and/or errors is used to try to produce a smoother stepsize sequence.

The stepsize control loop can be regarded as consisting of a process and a controller (Figure 3). The process has one input: the stepsize h, and two outputs: the solution of the differential equation y and the error estimate r. The stepsize controller tries to keep the output r close to the tolerance *tol* using the stepsize h as control variable. The natural way to improve the

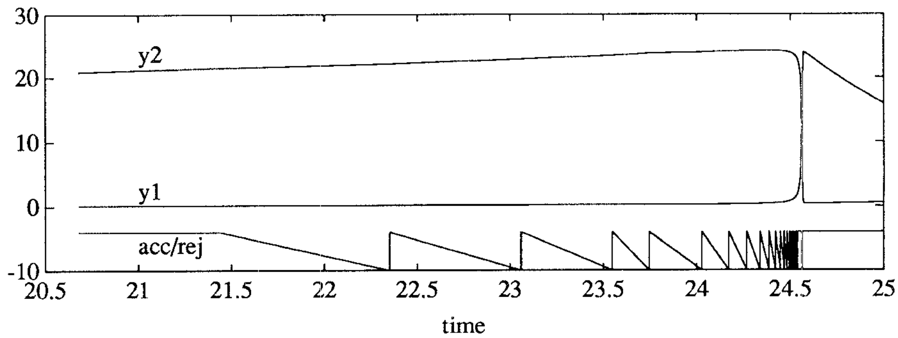

Fig. 2 A simulation of the Brusselator. The two upper curves are the state variables, while the lower third curve indicates whether the step in the integration method was rejected (level -10 in the plot) or accepted (level -4 in the plot). From $t = 21.0$ to $t = 24.5$ there are 24 accepted and 21 rejected steps.

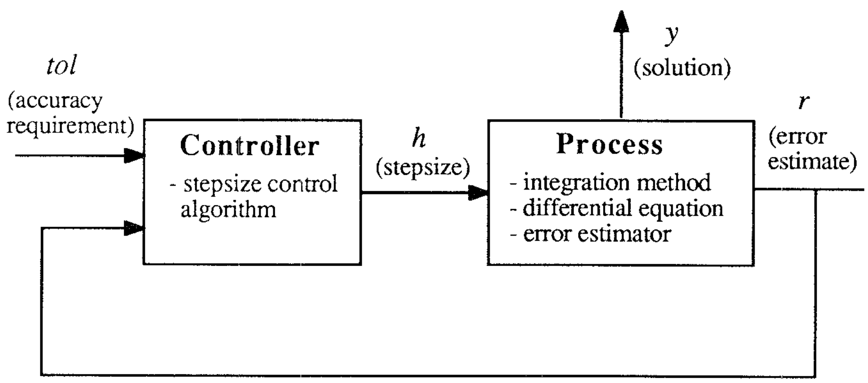

Fig. 3. Control system view of stepsize control.

stepsize control is to change the controller. By using elementary techniques from control theory, a controller on the form

$$h_n = \left(\frac{tol}{r_n}\right)^{k_I} \left(\frac{r_{n-1}}{r_n}\right)^{k_P} h_{n-1} \tag{2}$$

was suggested by Gustafsson et al. [5]. The controller has been tested for explicit Runge-Kutta on a variety of problems and was found to have good properties [5, 6]. Not only is the problem with stepsize oscillations resolved, but the controller in general produces smoother stepsize sequences. As a result, the error estimates show a more regular behavior.

To analyze the control system in Figure 3 a good process model is needed. The model must correctly describe the relation between h and r also in the case of an oscillatory stepsize sequence. In Section 2 such a model will be derived for explicit Runge-Kutta methods. Using the model it is straightforward to explain the properties of the control system both when using controller (1) as well as (2) (Sections 4 and 5). Moreover, the appropriate value of the parameters k_I and k_P in (2) depends on the integration method. An accurate process model allows for a more systematic way (Section 5 and Appendix B) of determining good values than the experimental approach taken by Gustafsson et al. [5].

2. MODELING THE PROCESS

An explicit m-stage Runge-Kutta method [7] applied to the initial-value problem $\dot{y} = f(t, y)$, $y(0) = y_0$ takes the following form

$$\begin{aligned}
\dot{Y}_1 &= f(t_n, y_n),\\
\dot{Y}_i &= f\left(t_n + c_i h_n, y_n + h_n \sum_{j=1}^{i-1} a_{ij} \dot{Y}_j\right), \quad i = 2 \ldots m,\\
y_{n+1} &= y_n + h_n \sum_{j=1}^{m} b_j \dot{Y}_j,\\
t_{n+1} &= t_n + h_n,\\
\hat{e}_{n+1} &= h_n \sum_{j=1}^{m} (b_j - \hat{b}_j) \dot{Y}_j,\\
r_{n+1} &= \begin{cases} \|\hat{e}_{n+1}\|, & \text{error per step (EPS)}\\ \|\hat{e}_{n+1}\|/h_n, & \text{error per unit step (EPUS)}. \end{cases}
\end{aligned} \tag{3}$$

The m-stage method supports two formulae of orders p and $p + 1$. The case when the coefficient set corresponding to order $p + 1$ is used to advance the solution is referred to as local extrapolation. Irrespective of which coefficient set is used for the integration, the error estimator is of order $p_e = p + 1$. The motive to use *EPUS* is to get the same "accumulated" global error for a fixed integration time regardless of the number of steps used.

For robustness, a mixed absolute-relative norm is used to form r. Two common choices are

$$\|\hat{e}\| = \max_i \left| \frac{\hat{e}_i}{\bar{y}_i + \eta_i} \right|, \quad \|\hat{e}\| = \sqrt{\sum_i \left(\frac{\hat{e}_i}{\bar{y}_i + \eta_i} \right)^2} \tag{4}$$

where $\bar{y}_i$ is (a possibly smoothed) absolute value of y_i, and η_i is a scaling factor for that component of y.

In explicit Runge-Kutta methods the $h - r$ relation shows two different dynamical behaviors: one in the asymptotic range (h small), and one when stability limits the stepsize. In both cases the influence of the differential equation may be regarded as an external perturbation. All explicit Runge-Kutta methods have qualitatively the same behavior, and we shall now derive the appropriate models for the two cases.

To obtain a linear model of the process near $h = 0$, consider the real scalar initial value problem

$$\dot{x} = \lambda(x - x_{stat}) \quad t \geq 0, \lambda < 0$$
$$x(0) = x_0. \tag{5}$$

The Runge-Kutta algorithm then yields the exact process

$$y_{n+1} = P(h_n\lambda)y_n, \quad \hat{e}_{n+1} = E(h_n\lambda)y_n \tag{6}$$

where $P(h_n\lambda)$ and $E(h_n\lambda)$ are polynomials in $h_n\lambda$, and $y_n = x_n - x_{stat}$.

The polynomial $E(h_n\lambda)$ takes the form $E(h_n\lambda) = \kappa_0(h_n\lambda)^{p_e} + \kappa_1(h_n\lambda)^{p_e+1} + \dots$, and hence the error estimate

$$r_{n+1} = \|\phi_n\| h_n^k, \quad \phi_n = y_n\lambda^{p_e}(\kappa_0 + \kappa_1 h_n\lambda + \dots), \tag{7}$$

with $k = p_e$ (EPS) or $k = p_e - 1$ (EPUS). Here ϕ_n is measured with the same norm as $\hat{e}$. For $|\kappa_0| \gg |\kappa_1 h_n \lambda|$ the process is described well using a *static* model. The coefficient vector ϕ_n is varying along the solution y_n. It is also dependent on h_n, but the dependence is weak as long as $|\kappa_0| \gg |\kappa_1 h_n\lambda|$.

By regarding $\log h$ as process input and $\log r$ as process output, the model is turned into an affine relation. Using the forward shift operator q, (7) yields

$$\log r_n = G_{p1}(q)\log h_n + q^{-1}\log\|\phi_n\|, \quad G_{p1}(q) = kq^{-1}. \tag{8}$$

Thus, the process is just a constant gain k, depending on the order of the error estimator in the integration method, and a disturbance $\log\|\phi_n\|$ depending on the properties of the differential equation and its solution (cf., Figure 4). The delay q^{-1} in the model is a consequence of the indexing conventions, i.e., the stepsize h_n is used to advance y_n to y_{n+1}, giving r_{n+1} as output. The model (8) takes the same form for the general problem $\dot{y} = f(t, y)$, $y(0) = y_0$, with ϕ formed from elementary differentials of f of order $p_e - 1$ and higher [6].

For the linear problem (5), $\phi \to 0$ as $t \to \infty$, and to keep r equal to *tol*, the stepsize controller will increase the stepsize. As $h_n\lambda$ increases the stepsize dependence of ϕ_n gets more pronounced, and consequently the process (8) behaves as if k had increased. For ϕ_n sufficiently small, $h_n\lambda$ reaches $\partial\mathcal{S}$, the boundary of the stability region $\mathcal{S} = \{h\lambda: |P(h\lambda)| \leq 1\}$ of the integration method. Increasing the stepsize beyond the critical value h_s makes the nonlinear difference equation system (6) unstable, and the stepsize is said to be limited by numerical stability. The behavior of (6) changes when $h_n\lambda$ approaches $\partial\mathcal{S}$, and the static linear model (8) no longer holds. Instead a new *dynamic* process model has to be derived. The derivation is similar in spirit to that of Hall [8, 9] and Hall and Higham [10].

A constant stepsize h_s leads to the stationary solution $|y_{n+1}| = |y_n|$, since $|P(h_s\lambda)| = 1$. Denote the steady state values by r_s and h_s, and

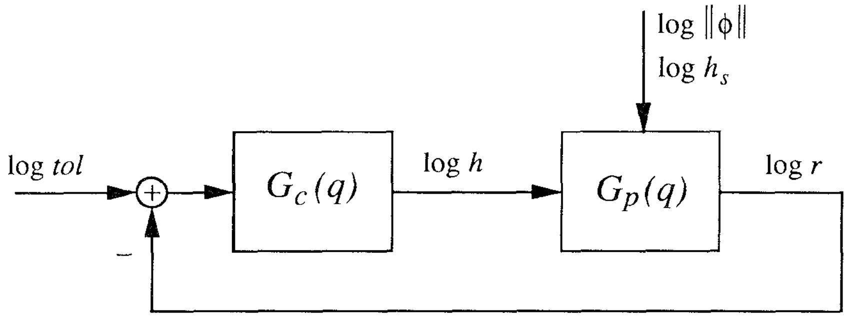

Fig. 4. The closed loop system where the transfer function $G_p(q)$ represents the process and $G_c(q)$ the controller. The differential equation acts as an external perturbation $\log\|\phi\|$ (or $\log h_s$).

consider small perturbations, i.e., $h_n = h_s(1 + \epsilon_n)$. Then

$$\begin{aligned}
\hat{e}_{n+1} &= E(h_n\lambda)y_n = E(h_n\lambda)P(h_{n-1}\lambda)y_{n-1} = E(h_n\lambda)\frac{P(h_{n-1}\lambda)}{E(h_{n-1}\lambda)}\hat{e}_n \\
&= E(h_s\lambda(1+\epsilon_n))\frac{P(h_s\lambda(1+\epsilon_{n-1}))}{E(h_s\lambda(1+\epsilon_{n-1}))}\hat{e}_n \\
&\approx E(h_s\lambda)(1+\epsilon_n C_1)\frac{P(h_s\lambda)(1+\epsilon_{n-1}C_2)}{E(h_s\lambda)(1+\epsilon_{n-1}C_1)}\hat{e}_n \\
&\approx P(h_s\lambda)(1+\epsilon_n)^{C_1}(1+\epsilon_{n-1})^{-C_1+C_2}\hat{e}_n \\
&= P(h_s\lambda)\left(\frac{h_n}{h_s}\right)^{C_1}\left(\frac{h_{n-1}}{h_s}\right)^{-C_1+C_2}\hat{e}_n
\end{aligned} \tag{9}$$

where

$$C_1(h_s\lambda) = h_s\lambda\frac{E'(h_s\lambda)}{E(h_s\lambda)}, \quad C_2(h_s\lambda) = h_s\lambda\frac{P'(h_s\lambda)}{P(h_s\lambda)}.$$

At steady state y_n is small, i.e., $r_n = |\hat{e}_n|/x_n \approx |\hat{e}_n|/x_{stat}$ (or if $x_{stat} < \eta$, $r_n \approx |\hat{e}_n|/\eta$. Using EPS, and noting that on the stability boundary $\partial\mathcal{S}$, $|P(h_s\lambda)| = 1$, (9) can be written

$$\log r_n = G_{p2}(q)(\log h_n - \log h_s), \quad G_{p2}(q) = \frac{C_1 q + C_2 - C_1}{q(q-1)}. \tag{10}$$

The dynamical behavior of the process is governed by $G_{p2}(q)$, and the only influence of the differential equation is the external perturbation $\log h_s$ (cf. Figure 4). Using EPUS changes the relation (10), and instead

$$\begin{aligned}
\log r_n &= G_{p3}(q)(\log h_n - \log h_s), \\
G_{p3}(q) &= \frac{(C_1 - 1)q + C_2 - C_1 + 1}{q(q-1)}.
\end{aligned} \tag{11}$$

By changing the definitions of C_1 and C_2 to

$$C_1(h_s\lambda) = \mathrm{Re}\left(h_s\lambda\frac{E'(h_s\lambda)}{E(h_s\lambda)}\right), \quad C_2(h_s\lambda) = \mathrm{Re}\left(h_s\lambda\frac{P'(h_s\lambda)}{P(h_s\lambda)}\right) \tag{12}$$

the results (10) and (11) may be generalized to all linear systems dominated by either a real negative eigenvalue or a complex conjugate pair of eigenvalues having a negative real part. An eigenvalue is said to be dominating, if it is the first eigenvalue to reach $\partial\mathscr{S}$ when the stepsize is increased, and at the same time the other eigenvalues are well inside $\mathscr{S}$. Examples of this type of generalization can be found presented by Gustafsson [6] and Hall and Higham [10].

3. EXPERIMENTAL VERIFICATION OF THE PROCESS MODELS

The process models (8), (10), and (11) are linearizations about $h = 0$ and $h = h_s$ and are valid in a neighborhood around these h-values. The structure of (8) indicates that as h increases (8) will still hold but behave as if the value of k had increased. Still, somewhere in the interval $h \in [0, h_s]$ there must be a transition from (8) to (11) or (10). The previous derivations do not explain what this transition looks like or where it takes place. By using system identification it is possible to partly answer these questions as well as verify the models (8), (10), and (11).

The nonlinear Robertson example (problem D2 [2]) was solved with different tolerances using DOPRI45, an order 4/5 explicit Runge-Kutta method [7], with local extrapolation and EPUS used for stepsize control. For each tolerance a transfer function from $\log h_n$ to $\log r_n$ was identified using the system identification toolbox in PRO-MATLAB [12]. The full identification procedure is described in Appendix A.

For DOPRI45 using local extrapolation the polynomials $P(z)$ and $E(z)$ are

$$P(z) = 1 + z + \frac{z^2}{2} + \frac{z^3}{6} + \frac{z^4}{24} + \frac{z^5}{120} + \frac{z^6}{600},$$
$$E(z) = -\frac{97z^5}{120000} + \frac{13z^6}{40000} - \frac{z^7}{24000}. \tag{13}$$

Let $\lambda_{\max}$ be the dominating most negative real eigenvalue of the Jacobian of the nonlinear differential equation. For $h_n\lambda_{\max} \approx -0.5$ the following model was obtained:

$$\log r_n = 4.25q^{-1}\log h_n. \tag{14}$$

Since the error estimator in DOPRI45 is of fifth order and EPUS was used, one would expect the value 4.0 instead of 4.25. The discrepancy is explained by the fact that $h\lambda$ differs significantly from zero and $|\kappa_0| \not\gg |\kappa_0 h\lambda|$. As a result, one does not observe the asymptotic behavior but a slightly modified one. By assuming that the behavior of the nonlinear equation is completely governed by $\lambda_{\max}$ it is possible to analytically estimate the modified behavior. One gets

$$\hat{e}_{n+1} \approx E(h_n \lambda_{\max}) y_n, \quad r_{n+1} \approx \frac{|E(h_n \lambda_{\max})| \, \|y_n\|}{h_n}.$$

It should then hold that

$$\frac{\partial \log r_{n+1}}{\partial \log h_n} = h_n \frac{\partial \log r_{n+1}}{\partial h_n} \approx h_n \frac{\partial}{\partial h_n} \log\left(\frac{|E(h_n \lambda_{\max})| \, \|y_n\|}{h_n} \right)$$
$$= h_n \lambda_{\max} \frac{E'(h_n \lambda_{\max})}{E(h_n \lambda_{\max})} - 1 = C_1(h_n \lambda_{\max}) - 1. \tag{15}$$

For $h_n \lambda_{\max} = -0.5$ the formula (15) evaluates to 4.19, which is in good agreement with the value 4.25 obtained from identification, (14). As $h_n \lambda_{\max}$ increases, higher order terms in $E(z)$ will play a larger role. Hence (15) predicts the gain 4.51 for $h_n \lambda_{\max} \approx -1.6$, while the identification gives 4.60. Note that these discrepancies are due to our assumption that $\lambda_{\max}$ dominates the behavior and that in (15) we differentiate the approximation for r_{n+1}. The polynomial $E(z)$ is, however, sufficiently smooth to allow this operation.

As $h_n \lambda_{\max}$ is further increased it will approach $\partial \mathscr{S}$ where now a model of the form (11) is expected. Although not veiified by theoretical derivations the identification indicates a gradual change from (8) to (11). As an example consider $h_n \lambda_{\max} \approx -2.4$. For this value the identification resulted in

$$\log r_n = \frac{4.87q - 0.15}{q(q - 0.24)} \log h_n.$$

For DOPRI45 the negative real axis intersects $\partial \mathscr{S}$ at -3.31. At this point $C_1 = 5.85$ and $C_2 = 6.07$, and according to (11) one would expect the model

$$\log r_n = \frac{4.85q + 1.22}{q(q - 1)} \log h_n.$$

This is in almost perfect agreement with the identified model for $h_n \lambda_{\max} \approx -3.3$, see Table I.

4. THE CLOSED LOOP USING THE STANDARD CONTROLLER

Assuming ϕ constant (or slowly varying) in (8) and trying to make $r = tol$ in the next step, leads to the stepsize controller (1). A step is rejected if $r > \nu tol$ ($\nu \geq 1$), and to reduce the risk of rejection, the safety factor γ ($\gamma \leq 1$) is introduced. Typical values for ν and γ are 1.2 and 0.9, respectively. In addition, there is often a limit on how much the stepsize may increase in one step.

Table I. The Identified Transfer Functions from $\log h_n$ to $\log r_n$.

		k/q	$\frac{\beta_1 q+\beta_2}{q(q-\alpha)}$		
tol_0	$h_n\lambda_{max}$	k	β_1	β_2	α
10^{-2}	−3.3	–	4.85	1.23	1.00
10^{-3}	−3.3	–	4.85	1.23	1.00
10^{-4}	−3.3	–	4.86	1.18	1.00
10^{-5}	−3.1	–	4.88	0.84	0.75
10^{-6}	−2.4	–	4.87	−0.15	0.24
10^{-7}	−1.6	4.60	–	–	–
10^{-8}	−0.90	4.41	–	–	–
10^{-9}	−0.54	4.25	–	–	–

By regarding $\log h$ as control variable, the standard stepsize controller (1) can be expressed as

$$\log h_n = \frac{1}{k}\frac{q}{q-1}\left(\log\left(\gamma^k \cdot tol\right) - \log r_n\right), \tag{16}$$

which can be interpreted as a control structure known as an integrating controller [3, 5]. The set point of the controller is $\log(\gamma^k \cdot tol)$, and $\log h_n$ is the controller state. Note that the safety factor γ is equivalent to decreasing the set point from $\log(tol)$ to $\log(\gamma^k \cdot tol)$. Changing the set point changes the performance but does not alter the dynamics of the control system, and hence in the sequel γ will be assumed equal to 1.

In the standard controller (1) the integration gain is normally chosen as $1/k$. Here it will be kept as a free parameter k_I to investigate its influence on the closed loop system. Then, the transfer function from control error to stepsize for the controller (16), can be expressed as

$$\log h_n = G_{c1}(q)\left(\log tol - \log r_n\right), \quad G_{c1}(q) = k_I\frac{q}{q-1}. \tag{17}$$

For asymptotically small stepsizes the process is well approximated by (8). The same model structure is valid for both EPS and EPUS. The closed loop system (see Figure 4), combining (8) and (17), may be written

$$\log r_n = G_{tol}(q)\log tol + G_\phi(q)\log\|\phi_n\| \tag{18}$$

where

$$G_{tol}(q) = \frac{G_{c1}(q)G_{p1}(q)}{1 + G_{c1}(q)G_{p1}(q)} = \frac{kk_I}{q - 1 + kk_I},$$
$$G_{\phi}(q) = \frac{1}{q(1 + G_{c1}(q)G_{p1}(q))} = \frac{q-1}{q(q - 1 + kk_I)} \tag{19}$$

are transfer functions from, respectively, tolerance and disturbance to error estimate.

The characteristic equation (the denominator of $G_{tol}(q)$) has a root at $1 - kk_I$. The root determines the stability as well as the transient properties of the closed loop system. The difference operator $q - 1$ in the numerator of $G_{\phi}(q)$ will remove constant components in $\log \|\phi\|$ at a rate determined by the position of the root. Moreover, provided the closed loop system is stable, r will eventually approach tol since $G_{tol}(1) = 1$.

Choosing $k_I = 1/k$, as normally is done in the standard controller, places the root at the origin and makes the system as fast as possible. For this choice a constant disturbance is compensated in one step, but at the price of making r sensitive to higher frequency components in $\log \|\phi\|$. The position of the root is a tradeoff between response time and sensitivity, and hence the value of k_I is a design parameter and should not be regarded as given by $1/k$.

When numerical stability limits the stepsize the process changes character. Using the model for EPUS (11) together with the standard controller ($k_I = 1/k$), the closed loop system can be written (the case EPS is handled analogously)

$$\begin{aligned}
\log r_n &= G_{tol}(q)\log tol + G_{h_s}(q)\log h_s,\\
G_{tol}(q) &= \frac{G_{c1}(q)G_{p2}(q)}{1 + G_{c1}(q)G_{p2}(q)},\\
&= \frac{k_I((C_1 - 1)q + 1 - C_1 + C_2)}{q^2 + (-2 + k_I(C_1 - 1))q + 1 + k_I(1 - C_1 + C_2)},\\
G_{h_s}(q) &= -\frac{G_{p2}(q)}{1 + G_{c1}(q)G_{p2}(q)},\\
&= -\frac{(q-1)((C_1 - 1)q + 1 - C_1 + C_2)}{q(q^2 + (-2 + k_I(C_1 - 1))q + 1 + k_I(1 - C_1 + C_2))}.
\end{aligned} \tag{20}$$

Here, $G_{tol}(1) = 1$ and $G_{h_s}(1) = 0$. Therefore $\log h_s$, which is constant or slowly varying, will be removed and eventually $\log r$ will equal $\log tol$, provided the closed loop control system is stable.

The transient behavior as well as the stability of the control system is goverened by the roots of

$$q^2 + (-2 + k_I(C_1 - 1))q + 1 + k_I(1 - C_1 + C_2) = 0. \tag{21}$$

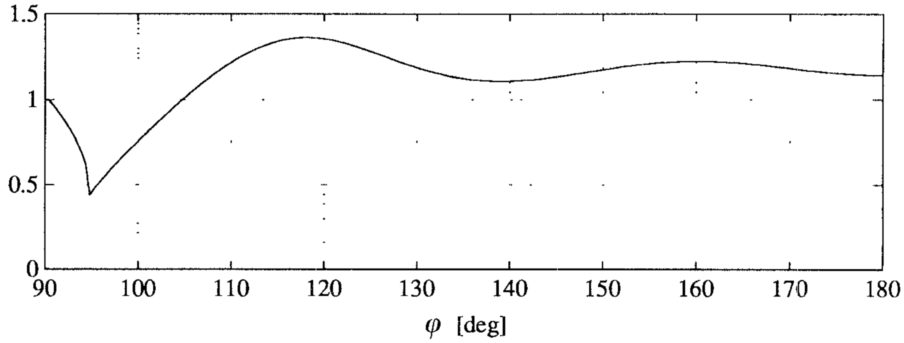

Fig. 5. The magnitude of the largest closed loop pole of the stepsize control loop when using DOPRI45 and the standard controller to solve a second order problem with complex eigenvalues, λ and $\bar{\lambda}$. The magnitude is plotted as a function of $\varphi = \arg(h_s\lambda)$. The closed loop system is stable only for φ close to $\pi/2$.

The system is stable if these roots are inside the unit circle. Hall [8, 9] derives another stability test consisting of an eigenvalue check for a 2 by 2 matrix. In the special case $k_I = 1/k$, the characteristic equation of that matrix equals the polynomial in (21). For the standard choice $k_I = 1/k$ the roots of (21) are often to be found outside the unit disc, resulting in an unstable closed loop system (see Figure 5 and Higham and Hall [11]).

When using the standard controller the closed loop system is almost always unstable for $h \approx h_s$. The instability causes the error to grow until the controller reduces the stepsize to keep the error below *tol*. The reduction of the stepsize moves $h\lambda$ inside $\mathscr{S}$ and the process changes behavior from (11) to (8), making the system regain stability. The error decreases and the controller will increase the stepsize, again placing $h\lambda$ on $\partial\mathscr{S}$. The cycle repeats itself creating an oscillatory stepsize sequence.

5. A NEW CONTROLLER

The properties of the closed loop system depend on the controller as well as on the process, and one may change either one to improve the behavior of the system. Higham and Hall [11] approach the problem by changing the process, namely, the integration algorithm. When constructing an explicit Runge-Kutta method there is some freedom in the choice of parameters. Normally this freedom is used to minimize error coefficients or to maximize the stability region of the method, but Higham and Hall exploit it to change C_1 and C_2 such that the closed loop system is stable when the standard controller is used.

It is our opinion that a better way to approach the problem is to change the controller. Then the freedom in choice of parameters in the method can be used to improve its numerical properties, while the stepsize control problem is solved by improving the stepsize controller.

The process models derived in Section 2 can be satisfactorily controlled with a controller on the form

$$G_{c2}(q) = k_I \frac{q}{q-1} + k_P = \frac{(k_I + k_P)q - k_P}{q-1}. \tag{22}$$

This controller is by no means arbitrary. The standard controller (17) is recognized as a commonly used control structure (discrete integral control) [3, 5]. Once this is realized, the modification to a discrete proportional-integral (or PI) controller as in (22) is straightforward. By manipulating the expression $\log h_n = G_{c2}(q)(\log tol - \log r_n)$, (22) can be rewritten as (2) [5]. From this expression it is clear that the new factor corresponds to taking the most recent development of r into account when deciding upon the next stepsize. It is also clear that this type of controller is trivial to implement in existing ODE codes.

Choosing the controller parameters k_I and k_P requires special attention. Their values are a compromise between stability and response time, and since C_1 and C_2 vary for different integration methods one cannot expect to find a single set of values that will be acceptable for all integration methods. Our tests indicate, however, that a reasonable first try is (see Appendix B)

$$k_I = \frac{0.3}{k}, \quad k_P = \frac{0.4}{k},$$

i.e., $k_I = 0.08$, $k_P = 0.10$ for DOPRI45 using EPUS and $k_I = 0.06$, $k_P = 0.08$ for EPS. A methodology to determine good parameter values is described in Appendix B.

There is a major difference between DOPRI45 and RKF45. For DOPRI45 $P(h_s\lambda) = 1$ (λ real), while for RKF45 $P(h_s\lambda) = -1$ (λ real). Hence, for a differential equation at stationarity, RKF45 will result in a numerical solution that oscillates around the true solution, since the sign of $\hat{e}$ alternates. If the stationary solution is nonzero the value of $|y|$ will change as the sign of $\hat{e}$ alternates. Through the relative norm (4) this affects r, causing both r and h to oscillate around their stationary values. The phenomena may be reduced by using smoothed values of y in (4).

The standard controller (1) (and in a sense also the PI controller (22)) is derived assuming $\log\|\phi\|$ constant or slowly varying. Consequently, the performance will not be acceptable for problems where $\log\|\phi\|$ changes rapidly (see Figure 2). The problem can be resolved using a controller that predicts changes in $\log\|\phi\|$. The controllers due to Watt [15] and Zonneveld [16] are of this type. Unfortunately, the $\log\|\phi\|$ prediction and the stabilization of (10) or (11) are conflicting objectives [6]. A predicting controller is therefore more suitable for integration methods with unbounded stability regions, or specialized problems and tolerances where the static model (8) always holds.

When a step is rejected the next step to be taken is a retry, and from the last attempt it is known what to expect ahead. The most likely reason for the rejected step is a major increase in the disturbance $\log\|\phi\|$. If the error from the rejected step is used to calculate $\log\|\phi_n\|$, and then a new stepsize h_n^* such that $\log r_{n+1}^*$ equals $\log tol$ is calculated, one obtains the standard controller,

$$h_n^* = \left(\frac{tol}{r_{n+1}}\right)^{1/k} h_n. \tag{23}$$

Since the previous step was rejected due to an increase in $\log\|\phi\|$, $\log\|\phi_n^*\|$ will generally be smaller than $\log\|\phi_n\|$. Therefore (23) is often a bit conservative.

Due to the structure of $\log\|\phi\|$ it is reasonable to assume that it will continue to increase during the steps succeeding the rejected step. Part of this increase can be anticipated by having the stepsize decrease appropriately after h_n^*. The state in the controller could be updated to achieve this end. At a rejected step h_n^* is calculated from (23) and used as the next stepsize. If h_n^* leads to an accepted step, the controller state is updated such that if the accepted step is perfect ($r_n^* = tol$) there will still be a stepsize decrease of the same factor as the one between h_n and h_n^*. If, on the other hand, h_n^* is rejected, (23) is used again.

The described strategy was used to solve the Brusselator [7]

$$\begin{aligned} \dot{y}_1 &= 1.0 + y_1^2 y_2 - (\beta + 1.0) y_1 & y_1(0) &= 1.3 \\ \dot{y}_2 &= \beta y_1 - y_1^2 y_2 & y_2(0) &= \beta \end{aligned}$$

with $\beta = 8.533$. In the time interval $t \in [21.0, 24.6]$ (see Figure 2) the number of rejected steps were decreased by almost 50% (from 39 to 21). Each rejected step is now normally followed by (at least) two accepted steps. The first accepted step is explained by (23) while the second is due to the special update of the controller state.

To summarize this section, an outline of the code implementing the new controller is presented in Listing 1 (see Figure 6). The controller is called after each step in the integration routine, and calculates the stepsize to be used in the next step. The variable x is the controller state, and as before, h is the stepsize, and r the corresponding error estimate. Occasionally, the error estimator may produce an unusually small (or large) value, thus advocating a very large stepsize change. For robustness the controller should (as usual) include some limitation on such big changes. Also, it is important to avoid overflow or underflow in expressions as $oldr/r$.

6. NUMERICAL TESTS

To demonstrate some of the properties of both the old and the new controller, some problems were simulated using DOPRI45 [7] with local extrapolation and EPUS. (A more extensive list of problems are simulated by Gustafsson et al. [5] and Gustafsson [6], although it is worth noting that the controller presented in Section 5 includes a better restart strategy after rejected steps.) The error was measured with the mixed absolute-relative 2-norm in (4), using $\eta_i = 0.1$, $\forall i$.

Fig. 6. Listing 1: An outline of the code needed to implement the new controller, including the restart strategy after rejected steps

if *current_step_accepeted* **then**

  if *previous_step_rejected* **then**

    $x := h \cdot h/x$

  endif

  $$x := \left(\frac{tol}{r}\right)^{k_I}\left(\frac{oldr}{r}\right)^{k_P} x$$

  $h := x$

  $oldr := r$

else

  $$h := \left(\frac{tol}{r}\right)^{1/k} h$$

endif

A simulation of $\dot{y} = -y + 1$, $y(0) = 1.1$ is shown in Figure 7, and demonstrates that the type of instabilities described in Section 4 occur also for very simple problems. As the solution approaches its stationary value the stepsize will increase. Eventually $h\lambda$ reaches $\partial\mathscr{S}$ and the process model changes from (8) to (11). For this case the standard stepsize controller fails to produce a smooth stepsize sequence, while the new controller performs well. For the new controller the stationary stepsize is approximately 3.3, and hence $h\lambda = -3.3$. This agrees well with theory since, for DOPRI45, $\partial\mathscr{S}$ intersects the negative real axis at -3.31.

Next we turn to the system used for the introductory example in Figure 1. The problem is a small control system consisting of a continuous time PID-controller [3] and a fourth order process. The variable y is the output from the process and y_{PID} is the output from the controller (and thus the input to the process). The system is described by

$$\begin{aligned} y &= \frac{1}{(\rho+1)^4} y_{PID} && \text{(process)} \\ y_{PID} &= k\left(y_r - y + \frac{1}{\rho T_i}(y_r - y) - \frac{\rho T_d}{\rho T_d/N + 1} y\right) && \text{(controller)} \qquad (24) \\ y_r &= 1 && \text{(reference signal)} \end{aligned}$$

with ρ being the differential operator. The parameter values $k = 0.87$, $T_i = 2.7$, $T_d = 0.69$ and $N = 30$ yield a PID-controller well tuned for the process.

Figure 8 shows some signals from the simulation of (24). The figure consists of six small plots where all signals are plotted as functions of time.

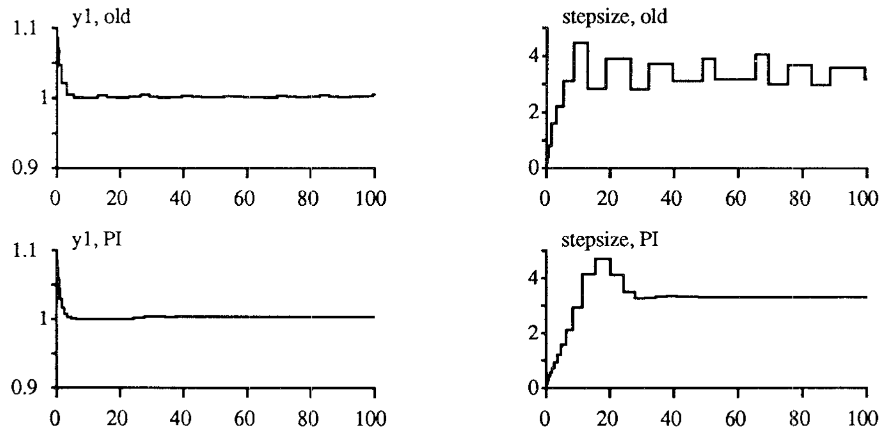

Fig. 7. Simulation of $\dot{y} = -y + 1$, $y(0) = 1.1$ with $tol = 10^{-3}$. At stationarity the old standard controller fails to produce a smooth stepsize sequence.

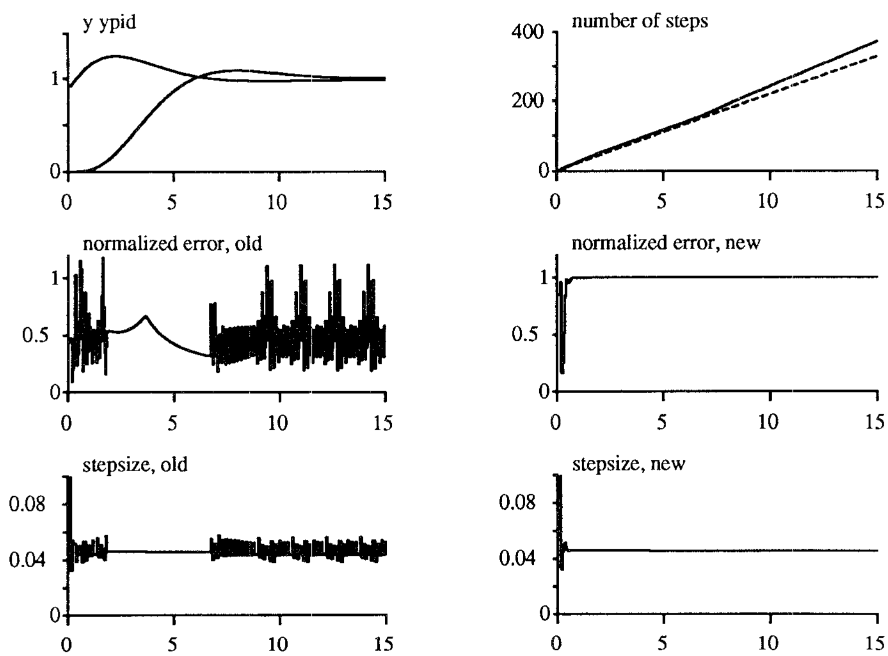

Fig. 8. Simulation of the small control system (24), $tol = 10^{-2}$.

The upper left plot shows the correct solution to the problem (y and y_{PID}). The upper right plot shows two curves corresponding to the work needed to solve the problem. It is the total number of integration routine calls for both the old standard controller (solid line) and the new controller (dashed line). Note that also the rejected steps are included to properly reflect the total work. The two plots in the middle show the estimated error r normalized with tol for the old (left) and the new (right) controller. The two lower plots compare the stepsize for the controllers. The last four plots include only succeeded steps (a step was rejected if $r > 1.2\,tol$).

The system (24) has four complex eigenvalues and two real. Five of the eigenvalues have a magnitude approximately equal to 1, while the sixth eigenvalue has $\lambda_6 \approx -40$. When solving (24) the transient corresponding to λ_6 dies out very fast. Consequently, the stepsize controller increases the stepsize, and soon $h\lambda_6$ is placed on $\partial\mathcal{S}$. The resulting irregular stepsize sequence excites the fast mode corresponding to λ_6. The error estimator fails to recover this mode properly and the solution produced is erroneous, as was demonstrated in Section 1 (see the plot of y_{PID} in Figure 1). The new controller produces the correct solution and reduces the number of rejected steps, thus decreasing the total amount of work needed to solve the problem by 20%.

The safety factor γ was chosen as 0.9 in the old controller used in the simulations. This makes the old controller aim for a lower error than the new controller. The effect is readily seen in Figure 8. One may argue that the safety factor lowers the set point making the old controller take more steps than the new one, and hence the comparison is unfair. However, if the safety factor is removed, the number of rejected steps (and hence the total work) increases drastically due to the irregularity of the stepsize sequence.

The standard controller was also equipped with a small dead-zone on stepsize change. Although normally not included in explicit methods, a small dead-zone was introduced in this comparison since it may sometimes increase the efficiency of the standard controller by preventing stepsize oscillations. This effect can be seen in the time interval $2 < t < 7$ in Figure 8. In spite of fixes like the safety factor and the dead-zone, the old controller cannot compete with the new.

7. CONCLUSIONS

Control theory provides efficient means to analyze the problem of stepsize control in numerical integration. It naturally separates an integration routine into two parts: the process (integration method, differential equation, and error estimator) and the stepsize controller. Hence, an integration method can be constructed for optimal numerical behavior, and then a fitting stepsize controller is designed.

To design the stepsize controller, a process model is needed. The static asymptotic relation, normally assumed, between the stepsize and the local truncation error is not always sufficient. When the stepsize is limited by numerical stability, a dynamic model has to be used. Such a model was derived and numerically verified for explicit Runge-Kutta methods.

Using the dynamic model, it is straightforward to analyze the standard stepsize controller. The analysis gives insight and clearly points out that the standard stepsize controller combined with a problem where numerical stability limits the stepsize leads to a locally unstable closed loop system.

The standard stepsize controller can be recognized as a commonly used control structure. A generalization of this structure is then natural, and using design techniques from automatic control, its parameters can be tuned such that good control is also achieved when numerical stability limits the stepsize. The new controller gives better overall performance at little extra expense.

Here only explicit Runge-Kutta methods were considered, and the proposed controller was of PI type. There is, however, nothing that limits the used methodology to these cases. Similar analytical techniques are applicable to other types of integration methods, and once a model is obtained it can be used to analyze and improve the stepsize control.

APPENDIX A. IDENTIFICATION OF A MODEL FOR DOPRI45

It is possible to verify the models derived in Section 2 using system identification. When simulating a differential equation the stepsize and error sequences are stored and used to fit a dynamical model between $\log h$ and $\log r$. If the stepsize and error sequences are taken from a time-interval where $\log\|\phi\|$ is relatively constant, its influence on the identification result can be removed. Consequently, the models will depend only on the tolerance and the integration method (DOPRI45 with EPUS in this case), and not on the differential equation.

The identification was done using problem D2 from the Enright et al. paper [2]

$$\begin{aligned} \dot{y}_1 &= -0.04y_1 + 0.01y_2y_3, & y_1(0) &= 1.0 \\ \dot{y}_2 &= 400y_1 - 100y_2y_3 - 3000y_2^2, & y_2(0) &= 0.0 \\ \dot{y}_3 &= 30y_2^2, & y_3(0) &= 0.0. \end{aligned} \tag{A.1}$$

The first 0.3 seconds of the solution to (A.1) and the stepsize sequences resulting from simulations with $tol = 10^{-2}, 10^{-3}, \ldots, 10^{-9}$ are shown in Figures 9 and 10, respectively. The new controller (22) described in Section 5 was used to prevent stepsize oscillations.

After an initial transient the stepsize stays essentially constant. The larger the value of *tol*, the larger this constant stepsize. This holds true for tolerances below 10^{-5}, while for larger values of *tol* the stepsize is sufficiently large to put $h\lambda_{\max}$ on $\partial\mathscr{S}$ for DOPRI45, i.e., $h\lambda_{\max} = 1.51 \cdot 10^{-3} \cdot (-2180) = -3.3$. The constant stepsizes indicate a constant $\log\|\phi\|$, and makes the problem ideal for identification.

The problem was solved with different values on *tol*. At $t = 0.1$, after the transient has died out completely, an excitation signal was added by perturbing $\log tol$ according to $\log tol = \log tol_0 + 0.05\,\Delta tol_n$. Here Δtol_n was a PRBS (pseudo-random binary signal [14]) sequence alternating between $+1$ and -1). The perturbation was small and the stepsize varied only a few percent around its stationary value. For each value of tol_0 the stepsize h_n and the error estimate r_n were recorded and stored. During the data logging there were no rejected steps. The experiment was done for $tol_0 = 10^{-2}, 10^{-3}, \ldots, 10^{-9}$. Figure 11 shows the stepsize and the error estimate recorded for $tol = 10^{-2}$.

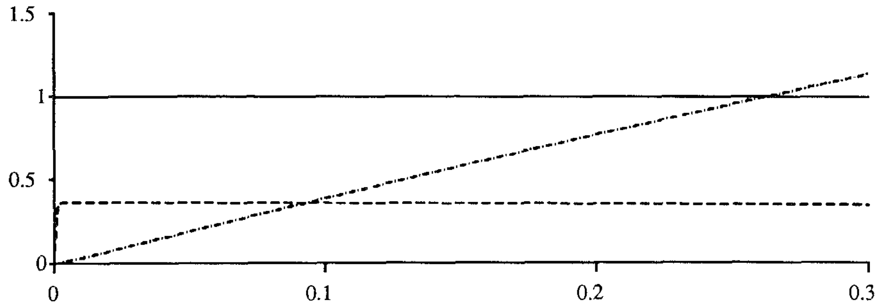

Fig. 9. The solution of the nonlinear problem (A.1), y_1 (solid line), y_2 (dashed line), and y_3 (dash-dotted line)

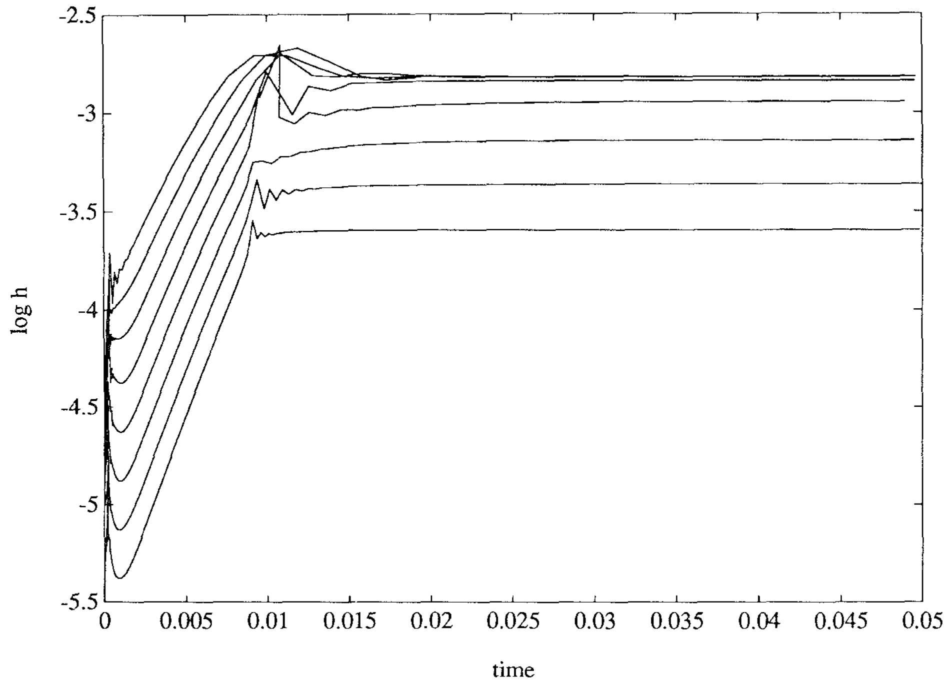

Fig. 10. The logarithm of the stepsize for different tolerances. The curves come in order, i.e , the lower one corresponds to $tol = 10^{-9}$, the second from the bottom corresponds to $tol = 10^{-8}$, and so on. The stepsize is practically constant for $t > 0\ 025$. This is true for $0.025 < t < 0.30$, although only $t < 0.05$ is plotted here For $tol = 10^{-4}, \ldots, 10^{-2}$ the stationary value of the stepsize is identical and the curves overlap, implying that the stepsize is limited by numerical stability.

The disturbance $\log \|\phi\|$ introduces a slow ramp in the data sequences. To remove its effect, a ramp fitted by least squares was subtracted from each data sequence. For each pair of data signals (stepsize sequence and error sequence) an ARMA-model from $\log h_n$ to $\log r_n$ was identified using the

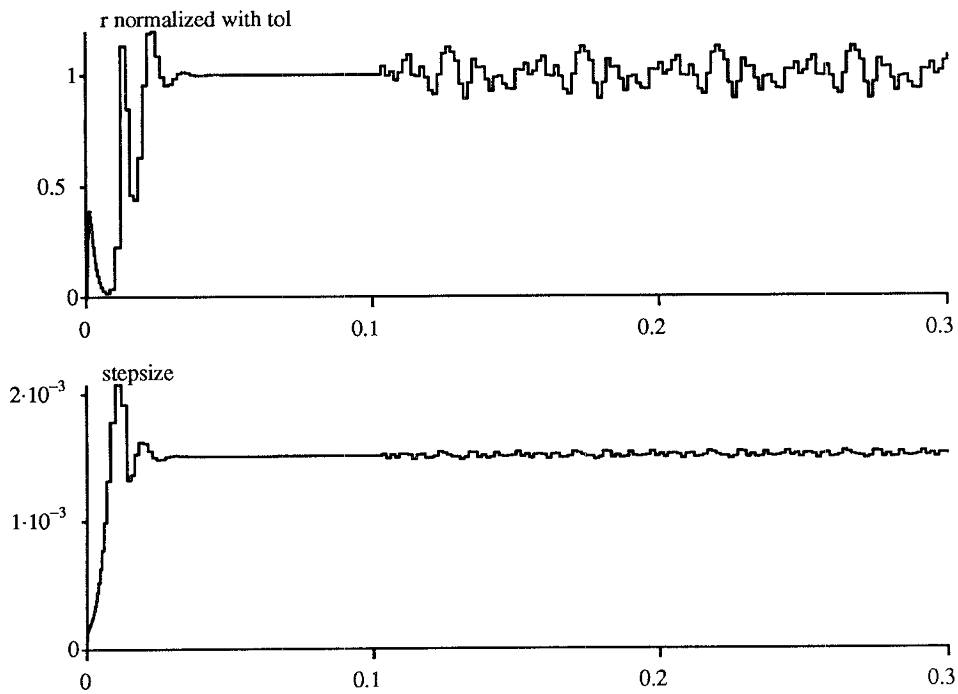

Fig. 11. Stepsize and error estimate recorded from the identification experiment with $tol = 10^{-2}$. The irregularities in the data sequences are caused by the perturbation of *tol*, and are not due to bad stepsize control.

identification toolbox in PRO-MATLAB [12]. The Akaike test and the statistics of the residuals were used to decide upon a correct model order [12, 14]. The identified transfer functions are listed in Table I.

As $h_n\lambda_{\max}$ increases to the value that puts $h_n\lambda_{\max}$ on $\partial\mathscr{S}$, the process gradually changes between the two models derived in Section 2. For $h_n\lambda_{\max}$ small and $h_n\lambda_{\max}$ on $\partial\mathscr{S}$, the models above agree very well with the theoretical results.

APPENDIX B. DETERMINING CONTROLLER PARAMETERS

The closed loop transfer function from log *tol* to log *r* takes the form

$$G(q) = \frac{G_c(q)G_p(q)}{1 + G_c(q)G_p(q)} \tag{B.1}$$

where $G_p(q)$ is the process and $G_c(q)$ is the controller (see Figure 4). Our task is the following: given $G_p(q)$ (e.g., $G_{p1}(q)$ in (8), $G_{p2}(q)$ in (10), or $G_{p3}(q)$ in (11)) choose $G_c(q)$ such that the difference equation relating log r and log *tol* is well behaved and the influence from $\log\|\phi\|$ (or $\log h_s$) on $\log r$ is minimized. This is the classical problem of feedback control [1, 3], and there exist many different methods to determine $G_c(q)$.

In our case the structure of $G_c(q)$ is already chosen ($G_{c2}(q)$ in (22)), and we are to determine k_I and k_P. It is worth noting that when designing a controller in a situation where a (almost) constant disturbance, i.e., $\log \|\phi\|$ or $\log h_s$, is to be eliminated, it is natural to choose a controller structure that includes integral action [1, 3].

The roots of the denominator of (B.1) determines the eigensolutions to the difference equation relating $\log r$ and $\log tol$. They also show up in the transfer functions from $\log\|\phi\|$ to $\log r$, and $\log h_s$ to $\log r$, and hence govern the behavior of the closed loop system. The system is stable if all the roots are inside the unit disc. Stable roots do not, however, necessarily give the system good properties. For instance, roots close to the unit circle correspond to eigensolutions with large time constants, and consequently the damping of the varying disturbance $\log\|\phi\|$ will be slow. On the other hand, roots close to the origin make the time constants small, and the system may be sensitive to noisy fluctuations in $\log\|\phi\|$.

The roots of the denominator of (B.1) are given by the solutions to

$$
\begin{aligned}
G_{p1}(q):\quad & q^2 + \left(-1 + k(k_I + k_P)\right)q - kk_P = 0 \\
G_{p2}(q):\quad & q^3 + \left(-2 + C_1(k_I + k_P)\right)q^2 + \left(1 + C_2(k_I + k_P)\right. \\
& \quad \left. - C_1(k_I + 2k_P)\right)q + k_P(C_1 - C_2) = 0 \qquad \text{(B.2)} \\
G_{p3}(q):\quad & q^3 + \left(-2 + (C_1 - 1)(k_I + k_P)\right)q^2 + \left(1 + C_2(k_I + k_P)\right. \\
& \quad \left. + (1 - C_1)(k_I + 2k_P)\right)q + k_P(C_1 - C_2 - 1) = 0
\end{aligned}
$$

The case $G_{p1}(q)$ is the most important, and k_I, k_P must be chosen such that these two roots have advantageous positions, e.g., stable, as fast as possible, sufficiently well damped. At the same time the roots for the case $G_{p2}(q)$ (or $G_{p3}(q)$) should be stable for as many C_1, C_2 values as possible. It is unfortunately not possible to achieve stability for any values on C_1, C_2, and we have to concentrate on the most likely ones. From our observations and from the structure of (12), it seems like it is often true that $C_1 \approx k$ and $C_2 \in [0, 2k]$ on most parts of $\partial\mathscr{S}$. In the case DOPRI45 $C_1 \in [5, 6]$ and $C_2 \in [0, 9]$ on $\partial\mathscr{S}$.

After having studied root positions for many different k_I, k_P values, we suggest

$$
k_I = \frac{0.3}{k}, \quad k_P = \frac{0.4}{k}, \qquad \text{(B.3)}
$$

as a reasonable tradeoff between root positions for the normal case $G_{p1}(q)$ and stability for relevant C_1, C_2 values. It is, however, important to fine tune the values given by (B.3) when using the controller with a new integration method. One way of doing this is to study root locus plots (for example, see Franklin et al. [3]).

Example B.1. Root locus plot. For DOPRI45 with local extrapolation using EPUS,

$$
G_{p3}(q) = \frac{4.85q + 1.22}{q(q - 1)},
$$

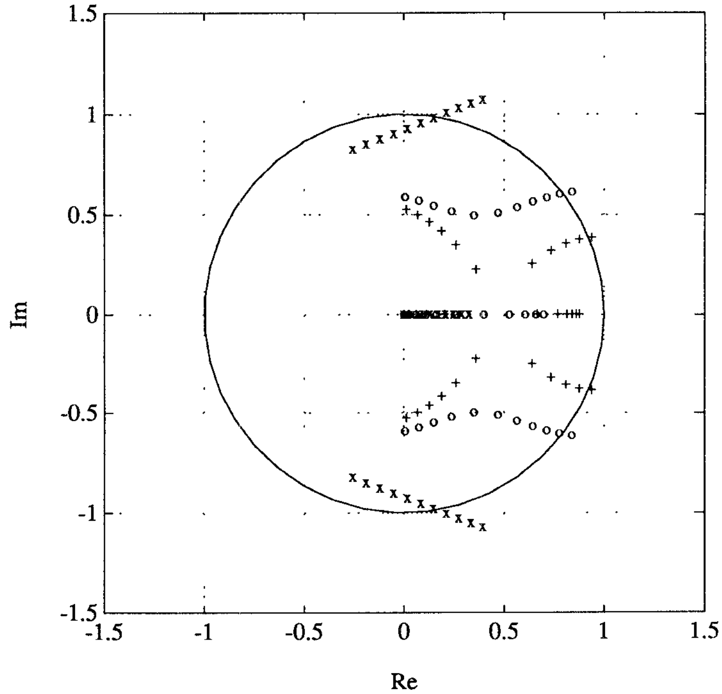

Fig. 12. Root locus from Example B.1. k_P is varied from 0 to 0.2 for three values of k_I: $k_I = 0.25$ ($\times$), $k_I = 0.12$ (0), and $k_I = 0.05$ (+). Small values of k_P correspond to the roots closest to the unit circle.

if evaluated at the point where $\partial \mathscr{S}$ intersects the negative real axis. In Figure 12 the roots of the denominator of (B.3) are plotted for $k_I = 0.25, 0.12$, and 0.05, varying k_P from 0 to 0.2. For small values of k_P, the system is unstable. Moreover, $k_I = 0.25$ results in roots close to the unit circle for every $k_P \in [0, 0.2]$. Using observations like this it is possible to conclude that $k_I < 0.15$, and $k_P \in [0.06, 0.2]$ results in reasonable roots for this $G_{p3}(q)$.

ACKNOWLEDGMENTS

Gustaf Söderlind has read several versions of the manuscript, and his comments and suggestions have greatly improved the paper. Constructive criticism from anonymous referees has also helped clarify the presentation.

REFERENCES

1. ÅSTRÖM, K. J., AND WITTENMARK, B. *Computer Controlled Systems, Theory and Design, 2nd ed.* Prentice-Hall, Englewood Cliffs, N.J., 1990, pp. 250–254, 271.
2. ENRIGHT, W. H. HULL, T. E., AND LINDBERG, B. Comparing numerical methods for stiff systems of ODE's. *BIT 15*, 1 (1975), 28–33.

3. FRANKLIN, G F, POWELL, J. D, AND EMAMI-NAEINI, A. *Feedback Control Systems* Addison-Wesley, Reading, Mass, pp. 95-100, 560
4. GEAR, C. W. *Numerical Initial Value Problems in Ordinary Differential Equations*. Prentice-Hall, Englewood Cliffs, N.J., pp 81-84.
5 GUSTAFSSON, K, LUNDH, M, AND SÖDERLIND, G. A PI stepsize control for the numerical solution of ordinary differential equations. *BIT 28*, 2 (1988), 270-287.
6 GUSTAFSSON, K. Stepsize control in ODE-solvers-analysis and synthesis Licentiate thesis TFRT-3199, Dept. of Automatic Control, Lund Inst. of Technology, Lund, Sweden.
7. HAIRER, E. NØRSETT, S. P., AND WANNER, G *Solving Ordinary Differential Equations, I, Nonstiff Problems*. Springer-Verlag, New York, pp. 112, 132, 167, 171.
8. HALL, G. Equilibrium states of Runge-Kutta schemes: Part I *ACM Trans. Math Softw. 11*, 3 (Sept 1985), 289-301
9. HALL, G. Equilibrium states of Runge-Kutta schemes: Part II *ACM Trans. Math Softw. 12*, 3 (*Sept*. 1986), 183-192.
10. HALL, G., AND HIGHAM, D. J. Analysis of stepsize selection for Runge-Kutta codes NA report 137, Univ of Manchester, 1987
11. HIGHAM, D. J., AND HALL, G Embedded Runge-Kutta formulae with stable equilibrium states. NA report 140, Univ of Manchester, 1987.
12. MOLER, C., LITTLE, J., BANGERT, S., AND KLEIMAN, S. *PROMATLAB, Users Guide*. The MathWorks, Inc., 1987.
13. SHAMPINE, L. F. Stiffness and nonstiff differential equation solvers. In *Numerische Behandlung von Differentialgleichungen*, L. Collatz, Ed. Information Series of Numerical Mathematics 27, Birkhauser Verlag, Basel, 1975, pp. 287-301
14. SÖDERSTRÖM, T, AND STOICA, P. *System Identification*. Prentice Hall, Englewood Cliffs, N.J., 1989.
15. WATTS, H. A Step size control in ordinary differential equation solvers. *Trans. Soc. Comput. Simul. 1*, 1, (May 1984), 15-25.
16. ZONNEVELD, J. A. Automatic numerical integration. Akademisch Proefschrift, Mathematisch Centrum, Amsterdam, 1964.

Received April 1989; revised November 1990; accepted November 1990