Digital Simulation of Continuous Stochastic Systems

Razevig, Vsevolod D

1977

Document Version:
Publisher's PDF, also known as Version of record

CODEN: LUTFD2/(TFRT-7121)/1-032/(1978)

DIGITAL SIMULATION OF CONTINUOUS STOCHASTIC SYSTEMS

VSEVOLOD D. RAZEVIG

Department of Automatic Control
Lund Institute of Technology
May 1977

Dokumentutgivare
Lund Institute of Technology
Handläggare Dept of Automatic Control

Dokumentnamn
REPORT

Dokumentbeteckning
LUTFD2/(TFRT-7121)/1-032(1977)

Utgivningsdatum

Ärendebeteckning

Författare
Razevig

Dokumenttitel och undertitel
Digital Simulation of Continuous Stochastic Systems

Referat (sammandrag)
Simulation algorithms for multivariable stochastic differential equations are given. Algorithms of different orders of convergence are derived using Picard iteration and mean square estimation of stochastic integrals. The algorithms are tested on a simple example. Both constant and variable integration step size are used. FORTRAN programs are included.

Referat skrivet av
author

Förslag till ytterligare nyckelord
Numerical Integration. ITO equations. Variable steplength.

Klassifikationssystem och -klass(er)

Indextermer (ange källa)

Omfång
32 pages

Övriga bibliografiska uppgifter

Språk
English

Sekretessuppgifter

ISSN

ISBN

Dokumentet kan erhållas från
Department of Automatic Control
Lund Institute of Technology
P O Box 725, S-220 07 LUND 7, Sweden

Mottagarens uppgifter

Pris

DOKUMENTDATABLAD enligt SIS 62 10 12

SIS-DB 1

Blankett LU 11:25 1976–07

DIGITAL SIMULATION OF CONTINUOUS STOCHASTIC SYSTEMS

V.D. Razevig

1. FORMULATION OF THE PROBLEM

A mathematical model of a dynamic system with stochastic perturbations can be described by the vector differential equation

$$dx = a(x,t)dt + b(x,t,)dw(t) \tag{1}$$

where x is an n-dimensional state vector,
w(t) a m-dimensional vector whose components are
a(x,t) is a n-vector of coefficients
and b(x,t) is n x m matrix of coefficient
It is assumed that both a(x,t) and b(x,t) satisfy Lipschitz conditions [3].

Equation (1) is considered as a stochastic differential equation in the sense of the stochastic integral [1]

$$I_\nu = \int_{t_0}^{t} \Phi(x(\tau),\tau)dw(\tau) = \tag{2}$$

$$= \lim_{\Delta\to 0} \sum_{i=0}^{N-1} \Phi((1-\nu)x(t_i) + \nu x(t_{i+1}),t_i)[w(t_{i+1}) - w(t_i)]$$

where $t_0<t_1<\ldots<t_N=t$, $\Delta=\max(t_{i+1}-t_i)$, $\Phi(x,t)$ is an arbitrary function of x and t, and the parameter ν lies within the interval $0\leq\nu\leq 1$. The Itô integral corresponds to $\nu=0$, the Stratonovich integral to $\nu=0.5$. One physical example which corresponds to intermediate values of ν is mentioned in [11].

The purposes of the report is to develop numerical algorithms of different order for integration of (1). Both constant and variable steplengths are considered. The integration interval $t\in[t_{min}, t_{max}]$ is divided into N segments $t_{min}=t_o<t_1<t_2<\ldots<t_N=t_{max}$ of sizes $h_r=t_{r+1}-t_r$.
When integrating equation (1) with constant steplength $h_r=h=const$, successively finer approximations to the same sample of noise w(t) were obtained by successive doubling

of N in such that the value of an approximation to the Wiener process w(t) at any point t_r equals the values of finer approximations at that time. When applying the variable steplength algorithms, the current step size h_r is an integer multiple of some smallest integration step size

$$h_{min} = \frac{t_{max} - t_{min}}{2^{K_{max}}} . \tag{3}$$

The current step size is thus equal to

$$h_r = \frac{t_{max} - t_{min}}{2^{K_r}} , \tag{4}$$

It is determined by the parameter K_r within the interval $K_{min} \leq K_r \leq K_{max}$.

The samples of the m-dimensional Wiener process $w(t)=\{w_j(t)\}$ are generated by computer as

$$w_j(t_{r+1}) = w_j(t_r) + \sqrt{h_{min}} \sum_{l=1}^{h_r/h_{min}} \zeta_{J_r+(l-1)m+j} \tag{5}$$

where $j=1,2,\ldots,m$, $z\zeta_1$ are Gaussian random numbers with zero mean and unite variance which are generated with the smallest sample interval h_{min}, J_r is the number of latest sample of the random number generator at the previous integration step t_r. See also Fig. 1.

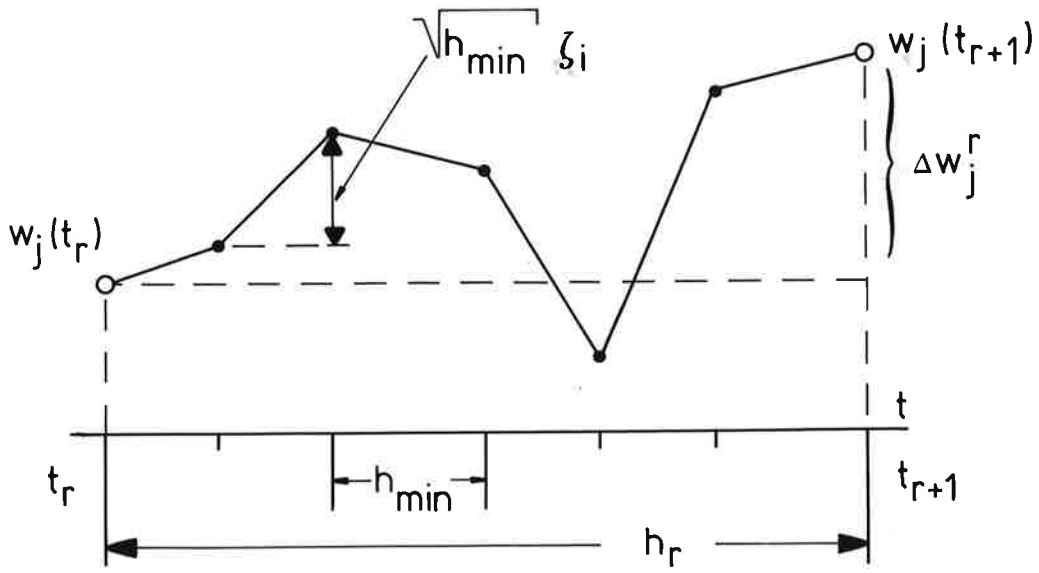


Fig. 1. Wiener process generator

2. SIMULATION ALGORITHMS AT ONE INTEGRATION STEP

Using Picard iterations [3] the following difference schemes are obtained

$$x^{[\ell+1]}(t_{r+1}) = x^{[\ell]}(t_r) + \int_{t_r}^{t_{r+1}} a(x^{[\ell]}(s),s)ds + \int_{t_r}^{t_{r+1}} b(x^{[\ell]}(s),s)dw(s) \quad (6)$$

where the indices in the square brackets indicates the number of iterations.

Using the zero approximation $x^{[0]}(t)=x(t_r)$ and a Taylor series expansion of the functions a_i and b_{ij} near the point $(x(t_r),t_r)$ a second approximation is obtained by retaining only the first-order derivatives of functions $a_i(x,t)$ and $b_{ij}(x,t)$. Hence

$$\begin{aligned} x_i^{[2]}(t_{r+1}) \approx\ & x_i(t_r) + a_i^r h_r + \frac{\partial a_i^r}{\partial t}\frac{h_r^2}{2} + \sum_{k=1}^{n}\frac{\partial a_i^r}{\partial x_k} a_k^r \frac{h_r^2}{2} + \\ & + \sum_{k=1}^{n}\frac{\partial a_i^r}{\partial x_k}\sum_{j=1}^{m} b_{kj}^r \int_{t_r}^{t_{r+1}} [w_j(s)-w_j(t_r)]ds + \sum_{j=1}^{m} b_{ij}^r [w_j(t_{r+1})-w_j(t_r)] + \\ & + \sum_{j=1}^{m}\left\{\frac{\partial b_{ij}^r}{\partial t} + \sum_{k=1}^{n}\frac{\partial b_{ij}^r}{\partial x_k} a_k^r\right\} \int_{t_r}^{t_{r+1}} (s-t_r)dw_j(s) + \\ & + \sum_{j=1}^{m}\sum_{k=1}^{n}\frac{\partial b_{ij}^r}{\partial x_k}\sum_{l=1}^{m} b_{kl}^r \int_{t_r}^{t_{r+1}} [w_l(s)-w_l(t_r)]dw_j(s) \quad (i=1,2,\ldots,n) \end{aligned} \quad (7)$$

where $h_r=t_{r+1}-t_r$ and the notations

$$a_i^r=a_i(x(t_r),t_r), \quad b_{ij}^r=b_{ij}(x(t_r),t_r)$$

has been introduced.

Equation (7) contains three stochastic integrals. The first two integrals are the same for any value of the parameter ν defined in (2). The Itô lemma [9] gives

$$[w_j(t_{r+1})-w_j(t_r)](t_{r+1}-t_r)=\int_{t_r}^{t_{r+1}}(s-t_r)dw_j(s)+\int_{t_r}^{t_{r+1}}[w_j(s)-w_j(t_r)]ds \tag{8}$$

Introduce the integral

$$\theta_j=\int_{t_r}^{t_{r+1}}[w_j(s)-w_j(t_r)]ds. \tag{9}$$

This integral can not be expressed in closed form. It is therefore approximated by its mean square estimate based on the assumption that the samples $w_j(t_r)$ and $w_j(t_{r+1})$ are known. Hence

$$\hat{\theta}_j = E\{\theta_j | w_j(t_{r+1}),\ w_j(t_r)\} = \frac{1}{2}\, h_r\, \Delta w_j^r \tag{10}$$

where

$$\Delta w_j^r = w_j(t_{r+1})-w_j(t_r). \tag{11}$$

The estimate $\hat{\theta}_j$ converges to θ_j for each realization of $w(t)$ as h_r is refined.

The diagonal elements ($l=j$) in the third stochastic integral at expression (7) can be calculated analytically, others ($l\neq j$) are estimated using the procedure outlived above. Hence

$$\hat{\psi}_{lj}^r = \int_{t_r}^{t_{r+1}}\Delta w_l(s)dw_j(s) = \begin{cases}\frac{1}{2}\,[\Delta w_j^r]^2-(\frac{1}{2}-\nu)h_r, & l=j,\\ \frac{1}{2}\,\Delta w_l^r\,\Delta w_j^r, & l\neq j.\end{cases} \tag{12}$$

Introducing (8), (10) and (12) into the approximation (7) retaining terms of different orders of magnitude, a set of algorithms for digital simulation of the stochastic differential equation (1)* are obtained.

2.1 First Order Algorithms

A first order algorithm is obtained from the general expression (7) by neglecting terms order $h_r\Delta w_j$, h_r^2 etc. The algorithm is given by

$$\hat{x}_i^{r+1} = \hat{x}_i^r + a_i^r h_r + \sum_{j=1}^{m} b_{ij}^r \Delta w_j^r + \sum_{j=1}^{m}\sum_{k=1}^{n} \frac{\partial b_{ij}^r}{\partial x_k} \sum_{l=1}^{m} b_{kl}^r \hat{\psi}_{lj}^r \qquad (13)$$

$$(i=1,2,\ldots,n)$$

where $\hat{\psi}_{lj}^r$ is defined in (12), $\hat{x}_i^r=\hat{x}_i(t_r)$, the $\hat{x}(t_r)$ marks the approximative value of exact solution $x(t_r)$.

The sample path convergence of the scheme (10) to the exact solution of equation (1) follows from [8] if the Itô stochastic integral is replaced by the general definition (2) and the new rules of stochastic integral calculations are considered.

If we neglect the last term of the right side of algorithm (13) the Euler scheme

$$\hat{x}_i^{r+1} = \hat{x}_i^r + a_i^r h_r + \sum_{j=1}^{m} b_{ij}^r \Delta w_j^r \qquad (i=1,2,\ldots,n) \qquad (14)$$

is obtained. This converges to the Itô solution of equation (1). It has an accuracy of order $\sqrt{h}$. Therefore, if $\nu>0$, algorithm (13) is the simplest algorithm for obtaining the numerical solution of the stochastic differential equation (1) in the sense of the general stochastic integral (2).

* We do not use higher than second order iterations (7), since we can not estimate more complicated stochastic integrals. If we replace higher order integrals by equivalent random values as is done in [5, 9], we get only statistical convergence, not the sample path.

2.2 Higher Order Algorithms

The algorithm order $h^{3/2}$ is obtained from the full expression (7):

$$\hat{x}_i^{r+1} = \hat{x}_i^r + a_i^r h_r + \sum_{j=1}^{m} b_{ij}^r \Delta w_j^r + \sum_{j=1}^{m} \sum_{k=1}^{n} \frac{\partial b_{ij}^r}{\partial x_k} \sum_{l=1}^{m} b_{kl}^r \psi_{lj}^r +$$

$$+ \frac{1}{2} h_r \sum_{j=1}^{m} \{\frac{\partial b_{ij}^r}{\partial t} + \sum_{k=1}^{n} \frac{\partial a_i^r}{\partial x_k} b_{kj}^r + \sum_{k=1}^{n} \frac{\partial b_{ij}^r}{\partial x_k} a_k^r\} \Delta w_j^r + \tag{15}$$

$$+ \frac{1}{2} h_r^2 \{\frac{\partial a_i^r}{\partial t} + \sum_{k=1}^{n} \frac{\partial a_i^r}{\partial x_k} a_k^r\}.$$

As mentioned above, we can not estimate more complicated stochastic integrals, than these given by (9) and (12), $\int (\Delta w(s))^2 ds$, for example. It is still possible to partly improve the accuracy of algorithm (15) retaining the higher derivatives of the functions $a_i(x,t)$ at the second iteration $x^{[2]}(t)$. Thus algorithms analogous to the forth-order Runge-Kutta (RK) scheme for deterministic equations [4] can be obtained as follows

$$\hat{x}_i^{r+1} = \hat{x}_i^r + \frac{1}{6}(K_{1i} + 2K_{2i} + 2K_{3i} + K_{4i}) - (\frac{1}{2} - \nu) \sum_{j=1}^{n} \sum_{k=1}^{m} \frac{\partial b_{ik}^r}{\partial x_j} b_{jk}^r h_r$$

$$(i=1,2,\ldots,n) \tag{16}$$

where

$$K_{1i} = a_i(\hat{x}^r, t_r) h_r + \sum_{j=1}^{m} b_{ij}(\hat{x}^r, t_r) \Delta w_j^r,$$

$$K_{2i} = a_i(\hat{x}^r + \tfrac{1}{2}K_1, t_r + \tfrac{1}{2}h_r) h_r + \sum_{j=1}^{m} b_{ij}(\hat{x}^r + \tfrac{1}{2}K_1, t_r + \tfrac{1}{2} h_r) \Delta w_j^r,$$

$$K_{3i} = a_i(\hat{x}^r + \tfrac{1}{2} K_2, t_r + \tfrac{1}{2} h_r) h_r + \sum_{j=1}^{m} b_{ij}(\hat{x}^r + \tfrac{1}{2} K_2, t_r + \tfrac{1}{2} h_r) \Delta w_j^r, \tag{17}$$

$$K_{4i} = a_i(\hat{x}^r + K_3, t_{r+1}) h_r + \sum_{j=1}^{m} b_{ij}(\hat{x}^r + K_3, t_{r+1}) \Delta w_j^r.$$

When the coefficients b_{ij} of equation (1) are small the scheme (16) has advantages compared with scheme (15) and (13) from a convergence-rate accuracy criterion. But for large b_{ij} the two schemes have approximately the same accuracy. A similar phenomenon was first mentioned in [2] and then in [4, p.190].

Another way to get RK type approximations of equation (1) is to apply the standard RK scheme [4] which converges to the Stratonovich solution of stochastic differential equation (1) (see [2], [6], [7], [8], [13]). Hence the solution of equation (1) in the sense of the general stochastic integral (2) can be obtained by applying the standard RK scheme to the Stratonovich equation, equivalent to equation (1) which is given by

$$dx_i^{Str} = a_{1i}(x,t)dt + \sum_{j=1}^{m} b_{ij}(x,t)dw_j(t) \qquad (i=1,2,\ldots,n) \qquad (18)$$

where

$$a_{1i}(x,t) = a_i(x,t) - (\frac{1}{2} - \nu) \sum_{j=1}^{n} \sum_{k=1}^{m} \frac{\partial b_{ik}}{\partial x_j} b_{jk}(x,t). \qquad (19)$$

The standard RK scheme to solve equation (18) is given by

$$\hat{x}_i^{r+1} = \hat{x}_i^r + \frac{1}{6}(K_{1i} + 2K_{2i} + 2K_{3i} + K_{4i}) \qquad (i=1,2,\ldots,n) \qquad (20)$$

where the coefficients $K_{1i},\ldots,K_{4i}$ are defined in (17) replacing the functions $a_i(x,t)$ by $a_{1i}(x,t)$ (19).

Our experiments show that the two RK type schemes (16) and (20) have approximately the same rate of convergence but scheme (16) demands less computations at each integration step.

3. VARIABLE STEP SIZE

The integration step size control and estimation of local truncation error are well developed, for ordinary differential equations without stochastic disturbances [4]. It would be desirable to have analogous techniques for the stochastic differential equations also. An empirical development is given below.

The main idea for the variable integration step size method is the following. Each basic step of size h_r is done twice, once as two steps of size $h_r/2$ and once as one step of size h_r. See Fig. 2.

If the result of one step of size h_r is $x^{(1)}(t_{r+1})$, and the result of two steps of size $h_r/2$ is $x^{(2)}(t_{r+1})$. The local truncation error is given by

$$\delta_{r+1} = \max_i |x_i^{(1)}(t_{r+1}) - x_i^{(2)}(t_{r+1})| . \qquad (21)$$

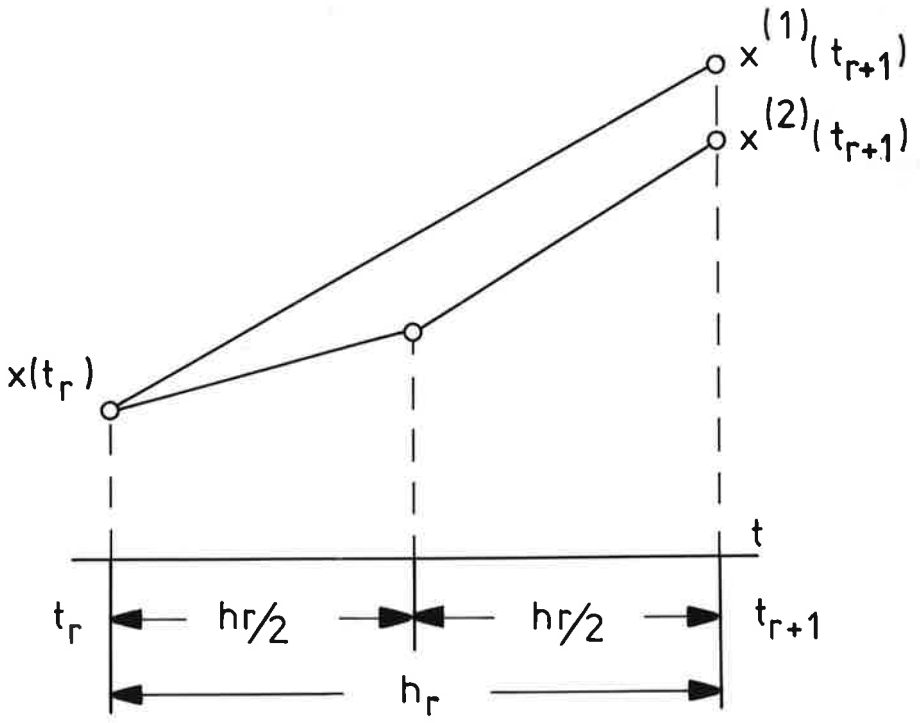


Fig.2. Variable step size

Compare δ_{r+1} with some limit error ε and modify steplength as follows:

1. If $\delta_{r+1} > \varepsilon$ restart from point t_r using the integration step length $h_r^{new} = h_r^{old}/2$. To get the same samples of Wiener process the state of the random number generator and the value of the Wiener process $w(t_r)$ at each initial point t_r are stored. For a more detailed description, see [10].
2. If $\varepsilon/10 < \delta_{r+1} < \varepsilon$ the next step is calculated using the step length $h_{r+1}=h_r$.
3. If $\delta_{r+1} < \varepsilon/10$ calculate the next step using step length $h_{r+1}=2h_r$.

This technique has been implemented for the RK method (20). See example in section 4.

4. EXAMPLES

4.1 A Simple First Order System

Consider the equation

$$dx = axdt + gxdw(t) \tag{22}$$

which has the Itô solution

$$x(t) = x(0)\ \exp\{(a - \tfrac{1}{2} g^2)(t - t_0) + g[w(t) - w(t_0)]\}. \tag{23}$$

Fig.3a shows the Itô solution of equation (22) at the point t=1 with initial condition x(0)=1 as a function of integration step size $h=1/2^K$ for various integration procedures with constant integration step length (parameters a= -1, g=1). Fig.3b shows the Itô solution of the equation

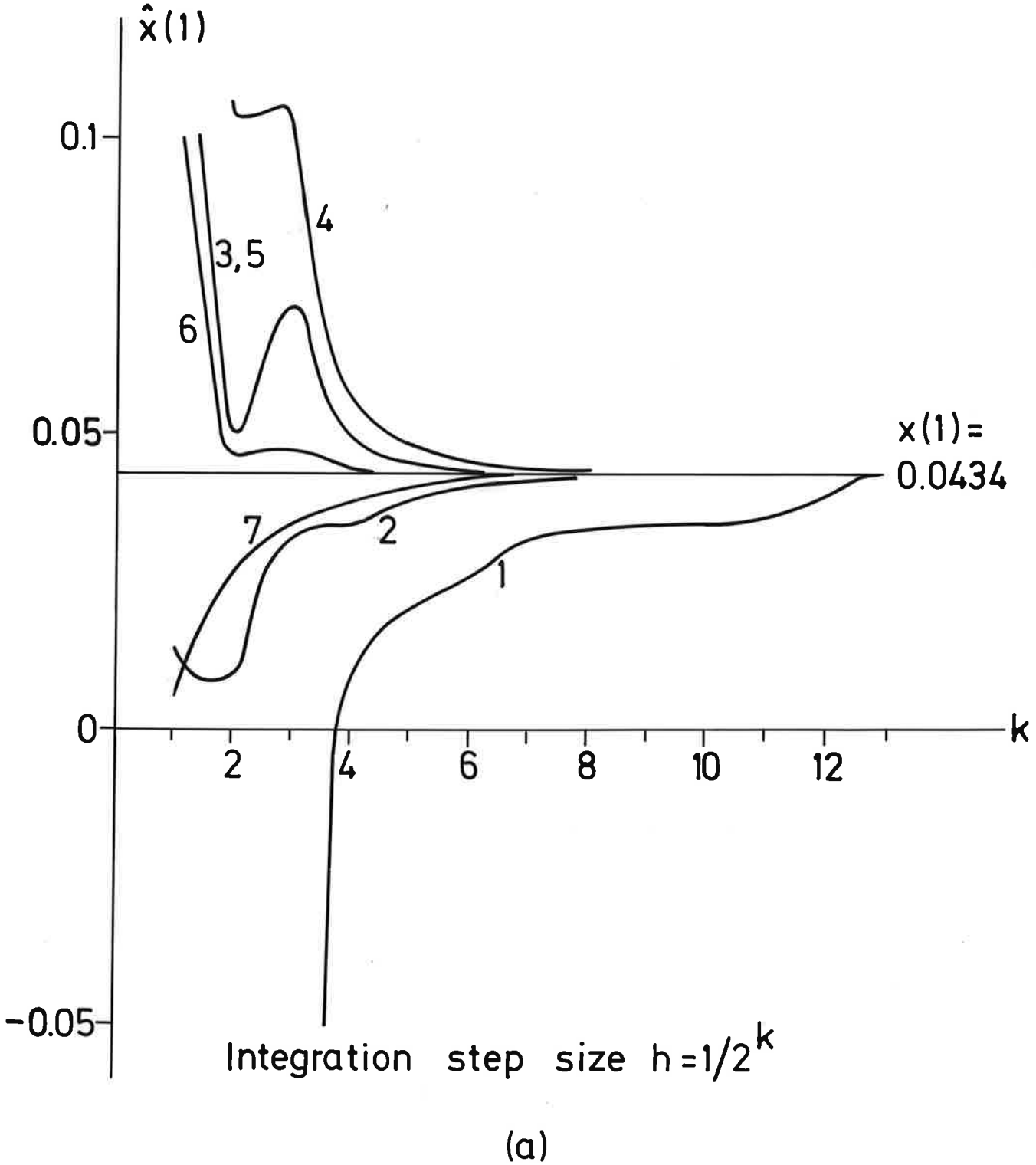


Fig. 3. - Simulation at the Itô equation (22) using different algorithms

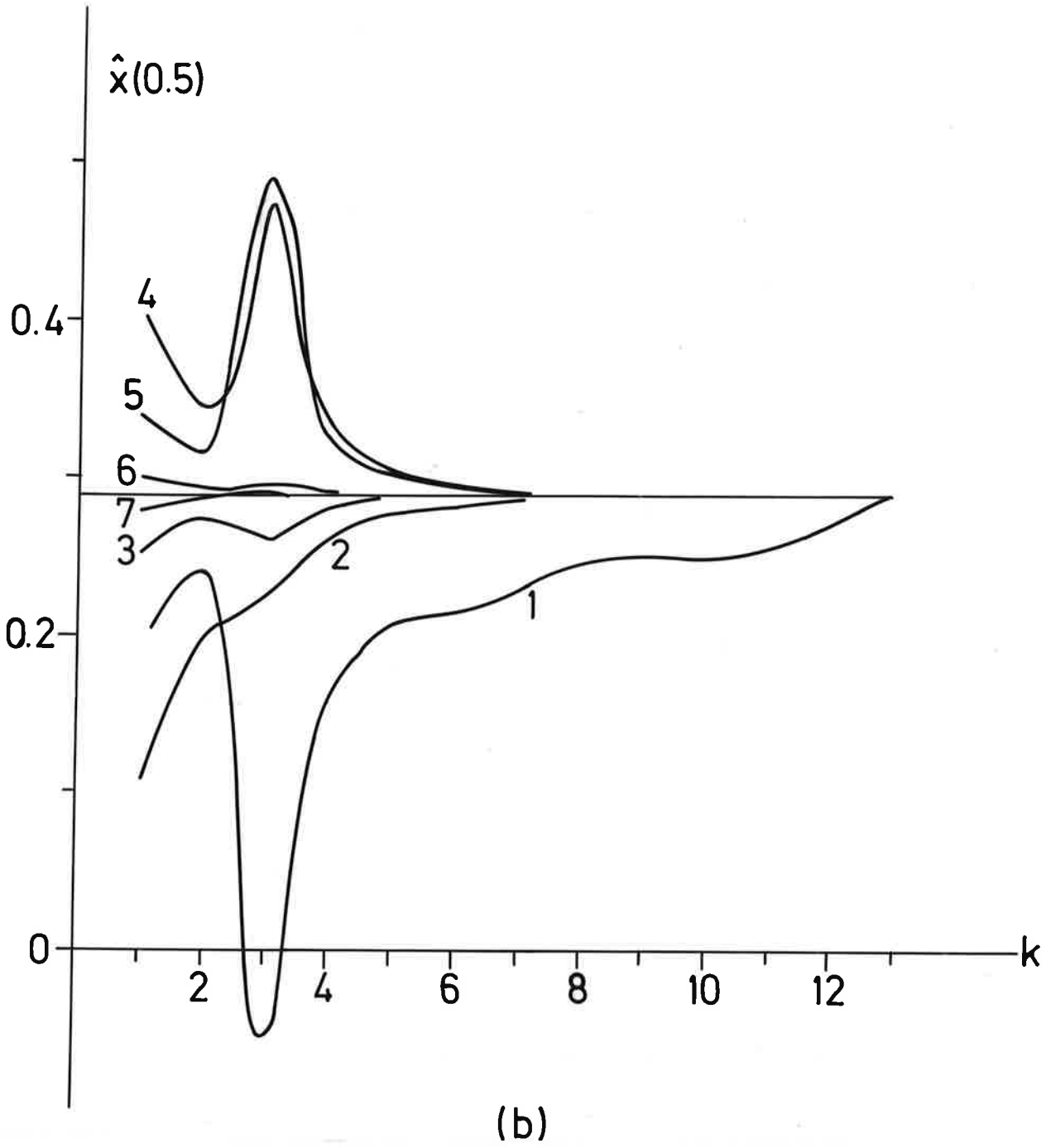


Fig. 3 Simulation of the Itô equation (24) using different methods.

1 - Euler method (14)

2 - first-order method (13)

3- second approximation (15)

4 - standard Euler-Cauchy method (second-order Runge-Kutta method) applied to the equivalent Stratonovich equation (18)

5 - Euler-Cauchy method analogues to (16)

6 - Standard Runge-Kutta method (20) applied to the equivalent Stratonovich equation (18)

7 - Runge-Kutta method (16)

$$dx = a \sin(x)dt + g \sqrt{|x|}\, dw(t) \qquad (24)$$

where $a = -1$, $g = 1$, $x(0) = 1$.

The relative rates of convergence were as follows:

quickest	- Runge-Kutta methods (16), (20),
	- second approximation (15),
intermediate	- Euler-Cauchy methods,
	- first-order method (13),
slowest	- Euler method (14).

4.2 Algorithms with variable Step Length

The accuracies of the integration methods with constant and variable step length applied to the Itô equation (22) are illustrated in Fig. 4. The picture shows how the accuracy defined as

$$\delta(t) = \left| \frac{x(t) - \hat{x}(t)}{x(t)} \right| \qquad (25)$$

depends on the number of steps integration. Note that $x(t)$ in expression (25) is the exact solution (23) of equation (22), and that $\hat{x}(t)$ is the numerical solution. Fig. 4 shows that RK method (20) with variable integration step length provides about 100 times higher accuracy with the same number of steps than RK method with constant step length. The variable step RK method is of course much more accurate than the Euler method (14) with constant step size. The advantage of the variable step size method increases with increasing accuracy.

4.3 A phase-locked loop

In the article [12] is discovered by practice that "the Euler integration is greatly superior to Runge-Kutta methods from a convergence-rate accuracy criterion". It is difficult to comment this statement since there was neither mathematical expression for Runge-Kutta method used (it is not obvious for stochastic equations) nor exact references. The simulation

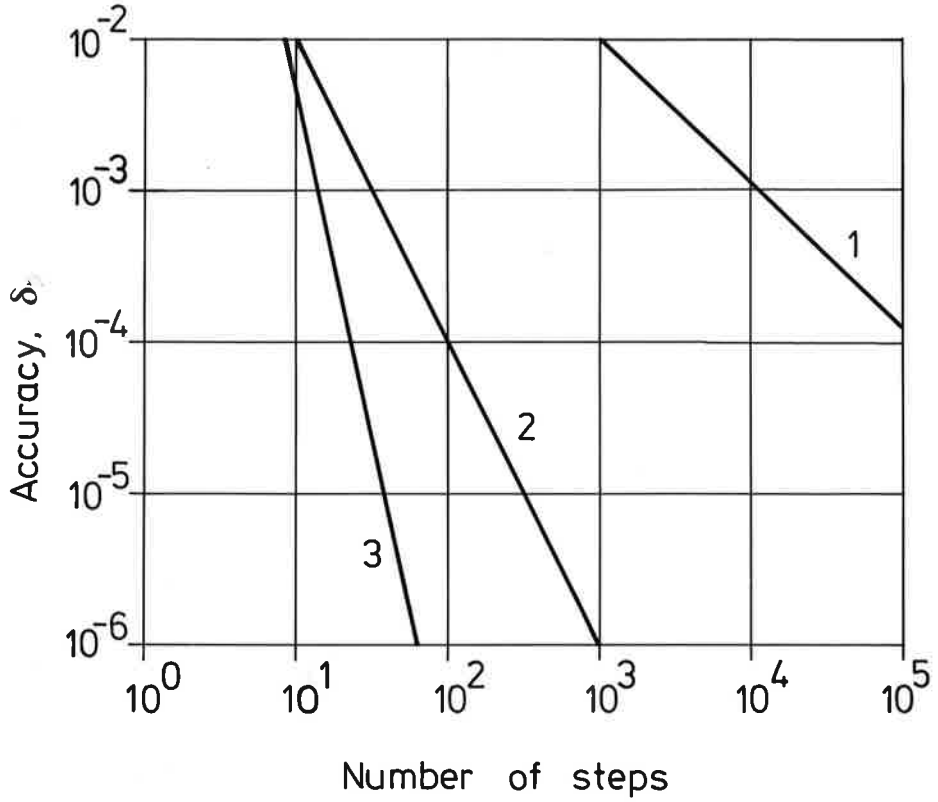


Fig. 4. - Accuracies of methods with constant and variable step size.

1 - Euler method (14) with constant step size

2 - Runge-Kutta method (16) with constant step size

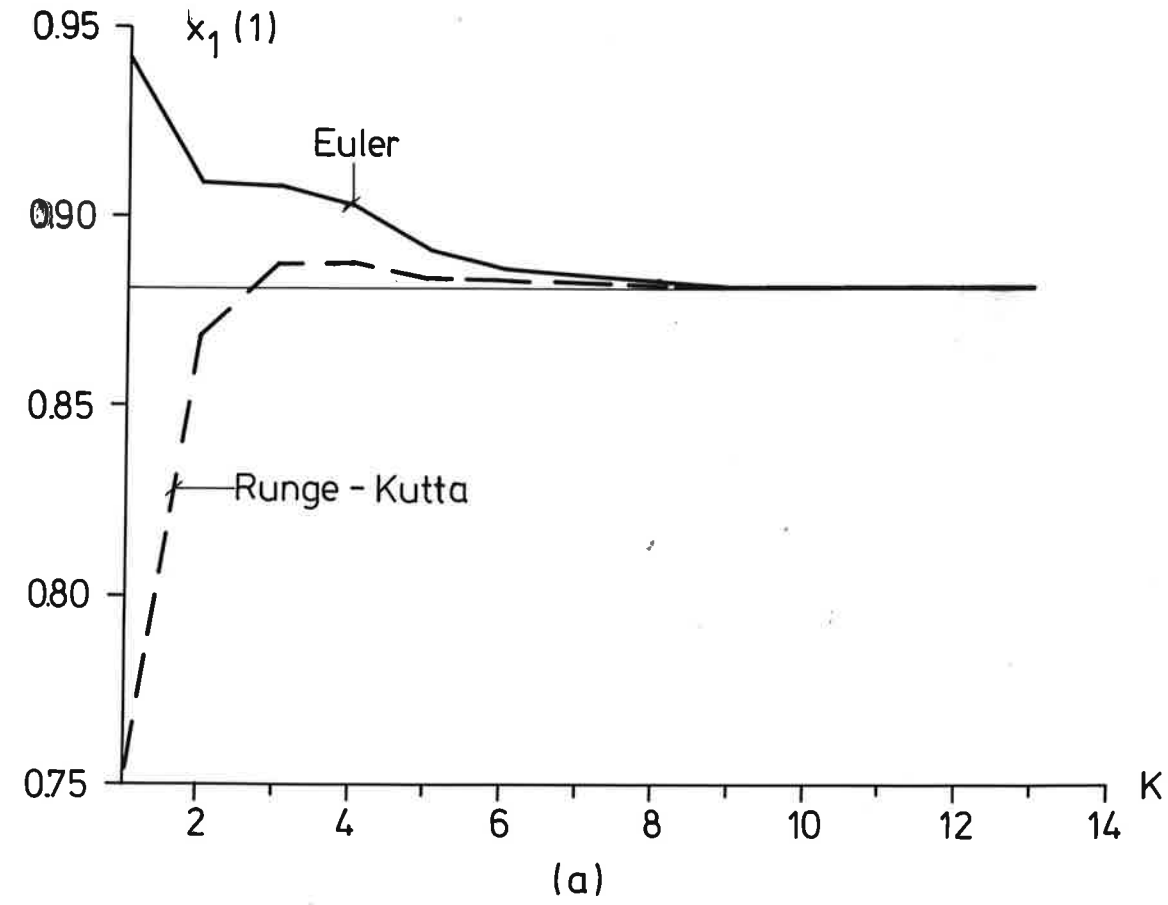
3 - Runge-Kutta method (16) with variable step size

of same phase-locked loop used in [12] has therefore been repeated. The equations are

$$\begin{cases} \dfrac{dx_1}{dt} = x_2, \\ \dfrac{dx_2}{dt} = -\sin(x_1) - \cos(x_1)\xi_1(t) - \sin(x_1)\xi_2(t) \end{cases} \tag{26}$$

where $\xi_1(t)$ and $\xi_2(t)$ are uncorrelated white random processes with unite spectral density. The Itô and Stratonovich solutions of equation (26) are the same since the term

$\Sigma\Sigma \dfrac{\partial b_{ik}}{\partial x_j} b_{jk}=0$. The Euler and Runge-Kutta methods (14) and (20) with constant step size where therefore used. The simulation results over the time interval $t\in[0.1]$ with initial conditions $x_1(0)=x_2(0)=1/4$ and with different integration step size $h=1/2^K$, $K=1,2...,14$ are shown at Fig. 5. Our results shows that the RK method (20) has a higher rate of convergence than the Euler method (14).



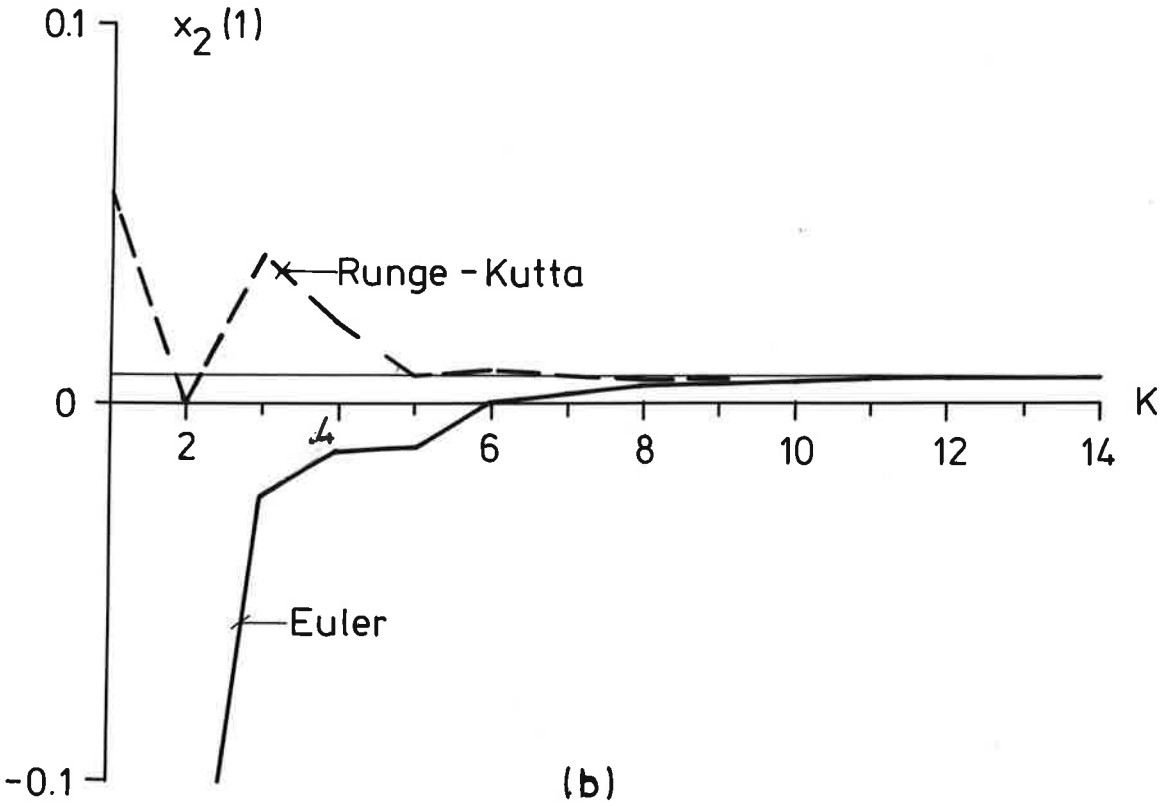


Fig. 5. - Simulation of the phase-locked loop.

CONCLUSION

A set of integration algorithms with different rate of convergence were developed for numeric solution of multi-dimensional stochastic differential equations (1) using both constant and variable integration step size. It was empirically shown by examples that the variable integration step size technique can be implemented for stochastic differential equations and taht it has advantages both with respect to accuracy and computation time. FORTRAN routines integrating equation (1) by Euler (14) and Runge-Kutta (20) methods with constant and variable integration step size were given in [10]. The programs used in this report are listed in the Appendix. In practice we recommend that the variable step size technique is used only with Runge-Kutta methods because of the slow convergence rate of the Euler method.

ACKNOWLEDGEMENTS

This work was done while the author was at the Department of Automatic Control, Lund Institute of Technology, under a grant from the Swedish Institute. I would like to thank the Swedish Institute for providing this opportunity and I would also like to thank Prof. K J Åström for many interesting discussions on integration of stochastic differential equations.

REFERENCES

1. K.J.Åström. Introduction to Stochastic Control Theory, Academic Press, New York, 1970.
2. I.G.Cumming. Computing aspects of problems in non-linear prediction and filtering. Ph. D. thesis, University of London, 1967.
3. J.l.Doob. Stochastic processes. Wiley, New York, 1953.
4. C.W.Gear. Numerical initial value problems in ordinary differential equations. Prentice-Hall, Inc. Englewood Cliffs, New Jersey, 1971.
5. C.J.Harris. Modelling, simulation and control of stochastic systems with applications in wastewater treatment. Int. J. Systems Sci., vol. 8, No. 4, 1977, pp. 393-411.
6. E.J.McShane. Stochastic calculus and stochastic models. Academic Press, New York, 1974.
7. N.N.Nikitin, S.V.Pervachev, V.D.Razevig. Solving stochastic differential equations of follow-up systems on digital computers. Automat. and Remote Control, No. 4, 1975, pp.133-137.
8. N.N.Nikitin, V.D.Razevig. Methods of digital simulation of stochastic differential equations and estimating of their accuracy. To appear in the "USSR J. Comput. Math. and Math. Phys.", No. 2, 1978.
9. N.J.Rao, J.D.Borwankar, D.Ramkrishna. Numerical solution of Itô integral equations. SIAM J. Control, vol. 12, No.4 , 1974, pp. 124-139.
10. V.D.Razevig. Simulation of non-linear stochastic differential equations. Dept. of Automatic Control, Lund Institute of Technology, Lund, Sweden, Report TFRT-7120 , 1977.
11. V.D.Razevig. Reduction of stochastic differential equations with small parameters and stochastic integrals. (Submitted to the Int. J. Control).

12. F.D.Taylor. On stochastic phase-lock loop solutions. IEEE Trans. on Communications, vol. COM-24, January 1976, pp. 138-140.

13. D.J.Wright. The digital simulation of stochastic differential equations. IEEE Trans. Automat. Control, vol. AC-19, February 1974, pp. 75-76.

14. Programming and Documentation Rules for Subroutine Libraries - Designed for the SCL. Eds Elmqvist, Tyssø, Wieslander, 1976.

Appendix

The FORTRAN program package for simulating the n-dimensional stochastic differential equation (1) contains three standard subroutines STOCH, STH, STEP, the main program, and at least two user subroutines SIDE and OUTP which are called on by standard subroutines, Tables 1-3. This package was tested on the computer PDP-15.

A.1 Main Program

The main program calls on the standard subroutine

```
CALL STOCH(XI,TMIN,TMAX,NX,NR,EPS,K,KMAX,KMIN,ISTEP,
METHOD,TP1,DP,NU,IS,W)
```

where the input arguments are:

XI - vector of initial conditions of state variables
TMIN - initial time of integration
TMAX - upper limit of integration time
NX - number of first-order equation n in (1)
NR - number of input Wiener processes m in (1)
EPS - local truncation error in the variable integration step length method, ε
K - defines the value of the integration step h (4) if ISTEP=0, otherwise K indicates the initial value of h
KMAX - indicates the minimum value of the integration step length h_{min} (3)
KMIN - indicates the maximum value h_{max}
ISTEP - if ISTEP=0, the integration interval $[t_{min}, t_{max}]$ is divided into 2^K equal segments of length h (4), otherwise the variable integration step method is used
METHOD - if METHOD=0, the subroutine obtains Itô solution by the Euler method, otherwise Stratonovich solution by the fourth-order Runge-Kutta method
TP1 - first point of time to output the solution by means of the user subroutine OUTP
DP - increment of time to output the solution
NU - initial state of the random number generator (odd number).

The argument IS is the output one which equals the number of integration steps. Vector W is the allocation vector.

In the main program the user has to define the input arguments and declare the real dimensions of vectors XI(NX) and W(7•NX+4•NR).

A.2 Standard Subroutines

Subroutine STOCH (Table 1) is the auxiliary subroutine for dynamic allocation of vectors in FORTRAN programs (see report [14]). This subroutine calls the basic subroutine STH (Table 2) which organizes the numerical integration with a constant, if ISTEP=0, or with variable step length otherwise. The subroutine STH also contains the Wiener process generator (5) which calls the library subroutine MCREDI.

Subroutine STH calls the standard subroutine STEP (Table 3) to integrate one step by the Euler method (14), if METHOD=0, or by the RK method (20) otherwise.

A.3 User Subroutines

A.3.1 Subroutine SIDE. Subroutine STEP calls on the user subroutine

```
SIDE(T+H,H,XH,R,DX)
```

where the input arguments are

T - current time, t_r
H - current integration step length, h_r, when using constant step technique, $h_r/2$ otherwise
X - initial vector $\vec{x}(t_r)$ on elementary integration interval (t_r, t_{r+1})
R - vector of Wiener process increments $\Delta\vec{w}_r$.

The user has to write the program to calculate the output vector argument DX of dimension NX in such a way.

For the Euler method (14) DX is the vector state variables increments, for the R-K method (20) it is the vector coefficient $\vec{K}_\ell$ (ℓ=1,2,3,4).

The vector DX contains coefficients of the stochastic differential equation (1). Let the n×1 vector A(I) = $\alpha_i(\vec{x}(t_r), t_r)$. Note that in many practical cases matrix $||\, b_{ij} \,||$ in equation (1) contains many zero elements. Therefore the user has to write down the n×m matrix $||\, b_{ij} \,||$ as a vector B(J) = $b_{ij}(\vec{x}_r, t_r)$ for non-zero elements b_{ij}. The dimension of B is equal to data of non-zero elements in matrix $||\, b_{ij} \,||$.

In the FORTRAN notation the i:th component of vector DX is equal to

$$DX(I) = A(I)*H+B(...)*R(1)+...+B(...)*R(NR), \qquad I = 1,2,...,NX. \tag{A.1}$$

Thus, in subroutine SIDE, the user has to define the vector coefficients A(I) and B(J) and write the expression (A.1) where H and R(J) are input arguments defined into basic subroutine STEP.

Due to allocation vector techniques the dimension of the vectors X, R, and DX can be described like this:

```
DIMENSION X(1),DX(1),R(1)
```

For vectors A and B the user has to point out their real dimensions.

A.3.2 Subroutine OUTP. Basic subroutine STH calls on the subroutine

```
OUTP(T,X,WIENER)
```

to output the current time T, the vector state variables X and the Wiener process WIENER.

All the arguments of subroutine OUTP are the input arguments defined into the basic subroutine STH.

Dimensions of the vector X and WIENER are described in such a manner:

```
DIMENSION X(1),WIENER(1).
```

```
001       C      NAME: STOCH
002       C      ----
003       C      SUBTITLE: SOLUTION OF THE NONLINEAR ITO OR STRATONOVICH STOCHASTIC
004       C      DIFFERENTIAL EQUATIONS
005       C
006       C      KEYWORDS:
007       C      --------
008       C      SOLUTION, SYSTEM OF NONLINEAR STOCHASTIC DIFFERENTIAL EQUATIONS,
009       C      ITO EQUATION, STRATONOVICH EQUATION
010       C
011       C      IMPLEMENTOR: VSEVOLOD D.RAZEVIG      DATE: 1977-05-19
012       C      -----------                          -----
013       C
014       C      INSTITUTE:
015       C      ---------
016       C      DEPARTMENT OF AUTOMATIC CONTROL
017       C      LUND INSTITUTE OF TECHNOLOGY, SWEDEN
018       C
019       C      ACCEPTED:                                    VERSION: 1
020       C      --------                                     -------
021       C
022       C=============================================================
023       C
024       C      PURPOSE
025       C      =======
026       C
027       C      ITO OR STRATONOVICH SOLUTION OF THE SYSTEM OF N FIRST-ORDER NONLINEA
028       C      STOCHASTIC DIFFERENTIAL EQUATIONS X'=A(X,T) + B(X,T)*W' WITH
029       C      CONSTANT OR VARIABLE LENGTH STEPS
030       C
031       C      USAGE
032       C      =====
033       C
034       C      PROGRAM TYPE: SUBROUTINE
035       C      ------------
036       C
037       C      ARGUMENTS:
038       C      ---------
039       C      STOCH(XI,TMIN,TMAX,NX,NR,EPS,K,KMAX,KMIN,ISTEP,METHOD,
040              1TP1,DP,NU,IS,W)
041       C
042       C      XI     - VECTOR OF INITIAL CONDITIONS OF STATE VARIABLES
043       C      TMIN   - INITIAL TIME OF INTEGRATION
044       C      TMAX   - UPPER LIMIT OF INTEGRATION TIME
045       C      NX     - NUMBER OF EQUATIONS
046       C      NR     - NUMBER OF INPUT WHITE NOISES
047       C      EPS    - LOCAL TRUNCATION ERROR
048       C      K      - INDICATE THE INITIAL VALUE OF THE INTEGRATION STEP
049       C               H=(TMAX-TMIN)/2**K
050       C      KMAX   - INDICATE THE MINIMUM VALUE OF THE INTEGRATION STEP
051       C               HMIN=(TMAX-TMIN)/2**KMAX
052       C      KMIN   - INDICATE THE MAXIMUM VALUE OF THE INTEGRATION STEP
053       C               HMAX=(TMAX-TMIN)/2**KMIN
054       C      ISTEP  - IF ISTEP=0, INTEGRATION INTERVAL DIVIDED INTO 2**K EQUAL
055       C               SEGMENTS OF LENGTH H, OTHERWISE ARE USED THE VARIABLE
056       C               INTEGRATION STEP
057       C      METHOD- IF METHOD=0, SUBROUTINE OBTAINS ITO SOLUTION BY EULER METHOD
058       C               OTHERWISE STRATONOVICH SOLUTION BY THE FOURTH-ORDER
059       C               RUNGE-KUTTA METHOD
060       C      TP1    - FIRST POINT OF TIME TO OUTPUT THE SOLUTION BY MEANS USER
061       C               SUBROUTINE OUTP
062       C      DP     - INCREMENT OF TIME TO OUTPUT THE SOLUTION
063       C      NU     - INITIAL STATE OF RANDOM NUMBER GENERATOR
```

```
064     C      IS    - NUMBER OF THE INTEGRATION STEPS
065     C      W     - ALLOCATION VECTOR HAVING DIMENSION 7*NX+4*NR
066     C
067     C      NOTES:
068     C      -----
069     C      1) USER COMPOSES THE MAIN PROGRAM WHICH CALLS ON SUBROUTINE STOCH
070     C      2) USER COMPOSES SUBROUTINE SIDE(T,H,X,R,DX)
071     C
072     C         INPUT ARGUMENTS:
073     C         ---------------
074     C         T   - CURRENT TIME
075     C         H   - CURRENT INTEGRATION STEP LENGTH
076     C         X   - CURRENT STATE VARIABLES VECTOR
077     C         R   - WIENER PROCESS INCREMENT VECTOR
078     C
079     C         OUTPUT ARGUMENT
080     C         ---------------
081     C         DX  - UNIT INCREMENT VECTOR EQUALS TO DX(I)=A(X,T)*H+B(X,T)*R(J)
082     C               (I=1,2,...,NX,  J=1,2,...,NR). NONLINEAR VECTOR COEFFICIENTS
083     C               A(X,T) AND B(X,T) ARE DESCRIBED BY USER. DIMENSION
084     C               OF A(X,T) EQUALG TO NX, MAXIMUM DIMENSION OF B(X,T)
085     C               EQUALS TO NX*NR.
086     C      3) USER COMPOSES SUBROUTINE OUTP(T,X,WIENER) TO OUTPUT
087     C         THE SOLUTION X(I) AND WIENER PROCESS WIENER(J).
088     C      4) SUBROUTINE STOCH CALLS ON INNER SUBROUTINES STH AND STEP
089     C      5) SUBROUTINE STH CALLS ON LIBRARY SUBROUTINE MCRED1 (RECTANGULAR
090     C         RANDOM NUMBER GENERATOR)
091     C
092     C      METHOD
093     C      ======
094     C
095     C      IN STH IS USED EULER OR FORTH-ORDER RUNGE-KUTTA METHOD BOTH WITH
096     C      CONSTANT AND VARIABLE INTEGRATION STEP LENGTH . THE CURRENT STEP
097     C      IS HALFING AND DOUBING IN SUCH A MANNER TO GET THE SAME SAMPLES
098     C      OF NOISE WHILE SOLVING THE EQUATION WITH DIFFERENT VALUES OF
099     C      EPS OR K.
100     C
101     C      REFERENCES:
102     C      ----------
103     C      1. D.J.WRIGHT, IEEE TRANS. ON AUTOMATIC CONTROL, V.AC-19, N 1, 1974
104     C      2. N.NIKITIN, S.PERVACHEV, V.RAZEVIG, AUTOMATION AND
105     C         REMOTE CONTROL, N 4, 1975.
106     C      3. C.W.GEAR. NUMERICAL INITIAL VALUE PROBLEMS IN ORDINARY
107     C         DIFFERENTIAL EQUATIONS, 1971.
108     C
109     C      CHARACTERISTICS
110     C      ===============
111     C
112     C      REVISIONS:
113     C      ---------
114     C
115     C-----------------------------------------------------------------------
116     C
117           SUBROUTINE STOCH(XI,TMIN,TMAX,NX,NR,EPS,K,KMAX,KMIN,ISTEP,METHOD,
118          1TP1,DP,NU,IS,W)
119     C
120           DIMENSION XI(1),W(1)
121     C
122     C
123           KW1=1
124           KW2=KW1+NX
125           KW3=KW2+NX
126           KW4=KW3+NX
127           KW5=KW4+NR
```

Table 1 (continued)

```
128             KW6=KW5+NR
129             KW7=KW6+NR
130             KW8=KW7+NR
131             KW9=KW8+NX
132             KW10=KW9+NX
133             KW11=KW10+NX
134       C
135       C     CALL ON THE BASIC SUBROUTINE STH
136       C
137             CALL STH(XI,TMIN,TMAX,NX,NR,EPS,K,KMAX,KMIN,ISTEP,METHOD,
138            1TP1,DP,NU,IS,W(KW1),W(KW2),W(KW3),W(KW4),W(KW5),W(KW6),W(KW7),
139            2W(KW8),W(KW9),W(KW10),W(KW11))
140             RETURN
141             END
```

Table 2

```
001       C
002       C     THIS IS BASIC SUBROUTINE TO SOLVE STOCHASTIC DIFFERENTIAL EQUATION
003       C
004             SUBROUTINE STH(X,TMIN,TMAX,NX,NR,EPS,K,KMAX,KMIN,ISTEP,METHOD,
005            1TP1,DP,NU,IS,X1,X2,XH2,WIENER,WIENE1,R,R1,S1,S2,S3,S4)
006       C
007             DIMENSION X(1),X1(1),X2(1),XH2(1),WIENER(1),WIENE1(1),R(1),R1(1),
008            1S1(1),S2(1),S3(1),S4(1)
009       C
010             DATA PI2/6.283185307/
011       C
012       C     DETERMINE ZERO INITIAL CONDITIONS FOR WIENER PROCESSES
013       C
014             DO 1 J=1,NR
015           1 WIENER(J)=0.0
016       C
017       C      ... FOR TIME T, SWITCH IS AND VARIABLE TPR RUNNIG
018       C      THE OUTPUTING SUBROUTINE OUTP
019       C
020             T=TMIN
021             IS=0
022             TPR=TP1
023             EPS1=EPS/10.0
024       C
025       C     DETERMINE THE MINIMUM AND MAXIMUME VALUES  OF INTEGRATION
026       C     STEP LENGTH
027       C
028             HMIN=(TMAX-TMIN)/2**KMAX
029             HMAX=(TMAX-TMIN)/2**KMIN
030             SHM=SQRT(HMIN)
031       C
032       C     CALL ON USER SUBROUTINE OUTP TO OUTPUT THE INITIAL CONDITIONS
033       C
034             CALL OUTP(T,X,WIENER)
035       C
036       C     THE BEGINING OF THE INTEGRATION.
037       C     DETERMINE THE CURRENT INTEGRATION LENGTH STEP
038       C
039           2 H=(TMAX-TMIN)/2**K
040             H2=H+H
041       C
042       C     STORAGE THE PRECEDING VALUE OF STATE NU OF RANDOM NUMBER GENERATOR
043       C     AND WIENER PROCESS FOR A RESTART
044       C
045             N1=NU
046             DO 3 J=1,NR
047             WIENE1(J)=WIENER(J)
048           3 R(J)=0.0
049       C
050       C     CALCULATE THE NUMBER OF CONSECUTIVE SAMPLINGS KR
051       C
052             KR=INT(H/HMIN+0.1)
053       C
054       C     GENERATE GAUSSIAN  INCREMENTS R(J) OF WIENER PROCESS AND
055       C     THE WIENER PROCESS  AT THE POINT T+H
056       C
057             DO 4 I=1,KR
058             DO 4 J=1,NR
059          31 CALL MCREDI(NU,REC1)
060             CALL MCREDI(NU,REC2)
061             IF(REC1.EQ.0.0)GOTO 31
062             GS=SQRT(-2.0*ALOG(REC1))*COS(PI2*REC2)
063           4 R(J)=R(J)+GS
```

Table 2 (continued)

```
064             DO 5 J=1,NR
065             R(J)=R(J)*SHM
066           5 WIENER(J)=WIENER(J)+R(J)
067       C
068       C     CALL ON SUBROUTINE STEP TO CALCULATE THE STATE VARIABLES
069       C     X1(I) ON THE FIRST STEP LENGTH H
070       C
071             CALL STEP(T,H,X,R,X1,NX,NR,METHOD,S1,S2,S3,S4)
072       C
073       C     IF ISTEP=0, PUT THE VALUES X1(I) TO THE CURRENT STATE VARIABLES
074       C     VECTOR X(I) AND GO TO THE BOTTOM OF SUBROUTINE
075       C
076             IF(ISTEP.NE.0)GOTO 7
077             T=T+H
078             DO 6 I=1,NX
079           6 X(I)=X1(I)
080             IS=IS+1
081             GOTO 16
082       C
083       C     IF ISTEP  NOT EQUALS TO ZERO, COMPUTE THE NEXT STEP
084       C     OF THE LENGTH H AND THE LARGER STEP OF THE LENGTH 2*H
085       C
086       C     GENERATE GAUSSIAN INCREMENTS OF WIENER PROCESS ON THE NEXT STEP R1(J)
087       C     AND CALCULATE WIENER PROCESS AT THE POINT T+2*H
088       C
089           7 DO 8 J=1,NR
090           8 R1(J)=0.0
091             DO 10 I=1,KR
092             DO 10 J=1,NR
093           9 CALL MCREDI(NU,REC1)
094             CALL MCREDI(NU,REC2)
095             IF(REC1.EQ.0.0)GOTO 9
096             GS=SQRT(-2.0*ALOG(REC1))*COS(PI2*REC2)
097          10 R1(J)=R1(J)+GS
098             DO 11 J=1,NR
099             R1(J)=R1(J)*SHM
100             WIENER(J)=WIENER(J)+R1(J)
101          11 R(J)=R(J)+R1(J)
102       C
103       C     CALL ON SUBROUTINE STEP TO CALCULATE XH2(I) ON THE LARGE STEP 2*H
104       C
105             CALL STEP(T,H2,X,R,XH2,NX,NR,METHOD,S1,S2,S3,S4)
106       C
107       C     CALL ON SUBROUTINE STEP TO CALCULATE X2(I) ON THE SECOND STEP H
108       C
109             CALL STEP(T+H,H,X1,R1,X2,NX,NR,METHOD,S1,S2,S3,S4)
110       C
111       C     CALCULATE THE LARGEST ERROR DM
112       C
113             DM=0.0
114             DO 12 I=1,NX
115             DE=ABS(XH2(I)-X2(I))
116             XA=ABS(X2(I))
117             IF(XA.GT.1.0)DE=DE/XA
118             IF(DE.GT.DM)DM=DE
119          12 CONTINUE
120       C
121       C     IS DM LARGER EPS?
122       C
123             IF(DM-EPS)13,13,20
124       C
125       C     IF DM<EPS, DETERMINE CURRENT TIME T=T+H AND CURRENT STATE VARIABLES
126       C     X(I)=X2(I). IF IN ADDITION DM<EPS/10  DOUBL THE LENGTH STEP
127       C
```

Table 2 (continued)

```
128          13 IF(DM.LT.EPS1)K=K-1
129             IF(K.LT.KMIN)K=KMIN
130          14 T=T+H2
131             IS=IS+2
132             DO 15 I=1,NX
133          15 X(I)=X2(I)
134       C
135       C     CHECK THE NECESSERITY TO CALL ON SUBROUTINE OUTP
136       C
137          16 IF(T-TPR)2,19,19
138          19 TPR=TPR+DP
139             CALL OUTP(T,X,WIENER)
140       C
141       C     IS T LARGER THAN TMAX?
142       C
143             IF(T+HMIN/2.0.LT.TMAX)GOTO 2
144             GOTO 25
145       C
146       C     IF DM>EPS,  REPEAT THE CALCULATIONS HALFING THE LENGTH STEP
147       C
148          20 K=K+1
149       C
150       C     IS K LARGER THAN KMAX?
151       C
152             IF(KMAX-K)21,22,22
153       C
154       C     IF K>KMAX, ELIMINATE K AND GO TO THE NEXT STEP
155       C
156          21 K=KMAX
157             GOTO 14
158       C
159       C     IF K LESS OR EQUAL KMAX RENEW THE STATE OF THE RANDOM NUMBER
160       C     GENERATOR AND WIENER PROCESSES AND REPEAT THE CALCULATIONS
161       C
162          22 NU=N1
163             DO 23 J=1,NR
164          23 WIENER(J)=WIENE1(J)
165             GOTO 2
166          25 RETURN
167             END
```

Table 3

```
001       C
002       C     THIS IS INNER SUBROUTINE TO PERFORM INTEGRATION ON ONE STEP.
003       C     IF METHOD=0, SUBROUTINE STEP OBTAINS ITO SOLUTION BY EULER
004       C     METHOD, OTHERWISE STRATONOVICH SOLUTION BY THE FORTH-ORDER
005       C     RUNGE-KUTTA METHOD.
006       C
007             SUBROUTINE STEP(T,H,X,R,XH,NX,NR,METHOD,S1,S2,S3,S4)
008       C
009             DIMENSION X(1),R(1),XH(1),S1(1),S2(1),S3(1),S4(1)
010       C
011       C     CALL ON USER SUBROUTINE SIDE TO DETERMINE THE INCREMENT OF
012       C     STATE VARIABLES ON THE STEP LENGTH H WITH INITIAL
013       C     CONDITIONS X(I) AT THE TIME T.
014       C
015             CALL SIDE(T,H,X,R,S1)
016       C
017       C     IS METHOD=0?
018       C
019             IF(METHOD.NE.0)GOTO 2
020       C
021       C     IF METHOD=0 CALCULATE THE NEXT SAMPLE XH(I) OF EULER SOLUTION
022       C     AT THE POIN T+H AND GO TO THE BOTTOM OF THE SUBROUTINE
023       C
024               DO 1 I=1,NX
025             1 XH(I)=S1(I)+X(I)
026               GOTO 7
027       C
028       C     IF METHOD NOT EQUAL TO ZERO CALCULATE THE NEXT RUNGE-KUTTA
029       C     COEFFICIENTS
030       C
031             2 DO 3 I=1,NX
032             3 XH(I)=X(I)+S1(I)*0.5
033               CALL SIDE(T+H*0.5,H,XH,R,S2)
034               DO 4 I=1,NX
035             4 XH(I)=X(I)+S2(I)*0.5
036               CALL SIDE(T+H*0.5,H,XH,R,S3)
037               DO 5 I=1,NX
038             5 XH(I)=X(I)+S3(I)
039               CALL SIDE(T+H,H,XH,R,S4)
040       C
041       C     DETERMINE THE NEXT SAMPLE XH(I) OF STRATONOVICH SOLUTION
042       C
043               DO 6 I=1,NX
044             6 XH(I)=(S1(I)+2.0*S2(I)+2.0*S3(I)+S4(I))/6.0+X(I)
045             7 RETURN
046               END
```