International Journal of Pharmaceutics, 73 (1991) 147–156
© 1991 Elsevier Science Publishers B.V. 0378-5173/91/$03.50
ADONIS 037851739100255B

IJP 02451

Curve fitting of stability data by personal computer. Software in pharmaceutics II

Sverre A. Sande and Jan Karlsen

Department of Pharmaceutics, Institute of Pharmacy, University of Oslo, P.O. Box 1068 Blindern, 0316 Oslo 3 (Norway)

(Received 17 January 1991)
(Modified version received 6 March 1991)
(Accepted 11 March 1991)

Key words: Stability testing; Reaction order; Mathematical model; Curve fitting; Computer program

Summary

A general mathematical model applicable for all reaction orders has been developed. A computer program written in C for use on personal computers performing curve fitting to the model is described. The time used by the program was tested with several datasets and found to be also acceptable with a minimum hardware configuration. The performance of the program was compared to that of a program for general non-linear regression running on a mainframe computer (SAS system on a VAX/VMS 8650). These tests showed that both methods produced the same results when the numbers of significant figures were the same.

Introduction

Predicting the shelf-life of a pharmaceutical formulation based upon accelerated studies is usually performed by assigning an order to the reaction and using the appropriate mathematical models for the calculation of the expiry date (e.g. when a 10% reduction of the active substance is expected) (Connors et al., 1986). Determination of the reaction order is usually performed by plotting the data or calculating according to the zero-, first- and second-order equations, thus determining the best fit (Thoma, 1978).

The purpose of this paper is to use a generalized equation for all orders (including orders between the integer orders), and to develop a computer program making the necessary calculations and graphically visualizing the results. This paper is based on computational principles corresponding to those utilized in a previous paper (Sande et al., 1989).

Calculations

The reactions taking place in the degradation of active substances in pharmaceutical preparations are usually very complex. It is therefore impossible to control all substances involved, and usually chemical analysis or stability testing of the preparation is limited to the active substance itself. The degradation of

Correspondence: S.A. Sande, Dept of Pharmaceutics, Institute of Pharmacy, University of Oslo, P.O. Box 1068 Blindern, 0316 Oslo 3, Norway.

the substance may therefore be described as:

$$\frac{\delta c}{\delta t} = -kC_s^x \tag{1}$$

where δc denotes the change in concentration over the time δt, k is a rate constant, x is the reaction order and C_s represents the concentration of the substance analyzed. Integration of Eqn 1 has to be performed under either of two conditions: (a) $x = 1$; (b) $x \neq 1$.

(a) $x = 1$: This is the usual first-order kinetics, and the integration is straightforward giving

$$C_s = C_0 e^{-kt} \tag{2}$$

where C_0 is the concentration at time 0.

(b) $x \neq 1$:

$$\int \frac{\delta c}{C_s^x} = \int - k\delta t \tag{3}$$

$$\frac{C_s^{(1-x)}}{1-x} = K_i - kt \tag{4}$$

where K_i is the integration constant. Substituting C_0 for C_s and 0 for t produces:

$$K_i = \frac{C_0^{(1-x)}}{1-x} \tag{5}$$

Inserting Eqn 5 into Eqn 4:

$$\frac{C_s^{(1-x)}}{1-x} = \frac{C_0^{(1-x)}}{1-x} - kt \tag{6}$$

Noting that this represents a general linearization of C_s vs t, we proceed solving Eqn 6 for C_s:

When $x > 1$ the general solution is:

$$C_s = \left(ktx - kt + C_0^{(1-x)}\right)^{\frac{1}{(1-x)}} \tag{7}$$

for $x < 1$ a general solution is only possible when

$$ktx - kt + C_0^{(1-x)} \geqslant 0 \tag{8}$$

since negative numbers raised to a decimal power do not produce a general real number solution.

Further:

$$tk(x-1) \geqslant -C_0^{(1-x)} \tag{9}$$

Changing from > to < since $x - 1$ is negative.

$$t \leqslant \frac{C_0^{(1-x)}}{k(1-x)} \tag{10}$$

At this time $C_s = 0$. A reasonable extension of the model would therefore be that $C_s = 0$ for times larger than the time given in Eqn 10.

Differentiation of Eqn 7 with respect to t produces negative values when $x < 1$ and $0 \leqslant t \leqslant C_0^{(1-x)}/k(1-x)$, and 0 for the upper boundary. When $x > 1$ the results are negative for all values of $t > 0$, and 0 only for a negative value of t.

The generalized model is therefore:

$$C_s = \begin{cases} 0 \leqslant x \leqslant 1: & \\ \quad \left(ktx - kt + C_0^{(1-x)}\right)^{\frac{1}{1-x}} & 0 \leqslant t \leqslant \dfrac{C_0^{(1-x)}}{k(1-x)} \\ \quad 0 & t > \dfrac{C_0^{(1-x)}}{k(1-x)} \\ x = 1: & \\ \quad C_0 e^{-kt} & t \geqslant 0 \\ x > 1: & \\ \quad \left(ktx - kt + C_0^{1-x}\right)^{\frac{1}{1-x}} & t \geqslant 0 \end{cases} \tag{11}$$

For all x the model equals C_0 for $t = 0$, and decreases monotonically for all $t > 0$.

Estimation of parameters

Although the best-fit parameters (x, k and C_0) may be evaluated by non-linear regression, a systematic variation of x from 0 up to a suitable order allows subsequent evaluation/presentation of several orders, and speeds up the calculation by enabling the use of weighted linear regression.

Linearization of the model for $x \neq 1$ is given in Eqn 6 while linearization for $x = 1$ is the usual logarithmation of Eqn 2:

$$\ln C_s = \ln C_0 - kt \tag{12}$$

Performing linear regression on transformed models necessitates weighting of the transformed data. The weighting is dependent upon whether or not the variance of the original data is constant over the entire data range. Since constant variance is the usual case for results from chemical analysis, the weighting performed is based on this assumption.

The weight imposed upon the transformed concentration data is the inverse of the transformed variance. Since the quotient between the variance of the transformed function and the concentration equals the quotient between the corresponding differentials:

$$\frac{\sigma(f(C_s))^2}{\sigma(C_s)^2} = \left(\frac{\delta(f(C_s))}{\delta(C_s)}\right)^2 \tag{13}$$

For $x \neq 1$:

$$\frac{\sigma(f(C_s))^2}{\sigma(C_s)^2} = \left(\frac{\delta(f(C_s))}{\delta(C_s)}\right)^2 = \left(\frac{\frac{1-x}{C_s^x}}{1}\right)^2 = \frac{(1-x)^2}{C_s^{2x}} \tag{14}$$

The transformed values are thus weighted by $C_s^{2x}/(1-x)^2$
For $x = 1$

$$\frac{\sigma(f(C_s))^2}{\sigma(C_s)^2} = \left(\frac{\delta(f(C_s))}{\delta(C_s)}\right)^2 = \left(\frac{\frac{1}{C_s}}{1}\right)^2 = \frac{1}{C_s^2} \tag{15}$$

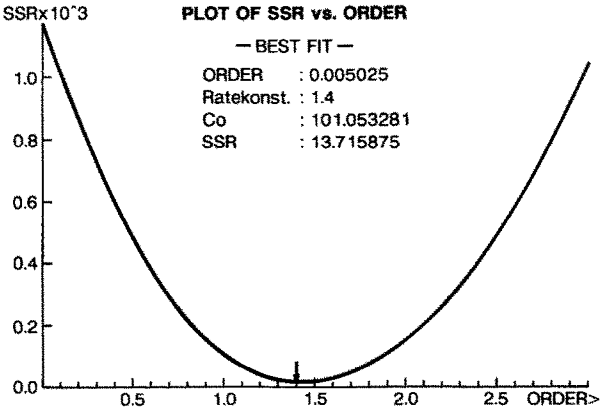
The transformed values are thus weighted by C_s^2.

If the relative variance is constant rather than the variance itself: $\sigma(C_s) = kC_s$, and the weights must be divided by C_s^2.

Program

An outline of the program is given in the Appendix. Following registration of the data, the rate constant (ratek), starting concentration (C_0) and sum of squared residuals (SSR) are calculated by weighted linear regression for all orders from zero to a specified maximum order (MAX_ORDER). The results are stored in an array for later presentation.

Weighting and transformation of the data according to the equations presented in the previous section (Eqns 14 and 15) are performed by the function LINFIT (see Appendix) prior to obtaining the slope and



```
— BEST FIT —
ORDER       : 0.005025
Ratekonst.  : 1.4
Co          : 101.053281
SSR         : 13.715875
```

Fig. 1. Screen dump showing the distribution of SSR as a function of reaction order for a sample data set.

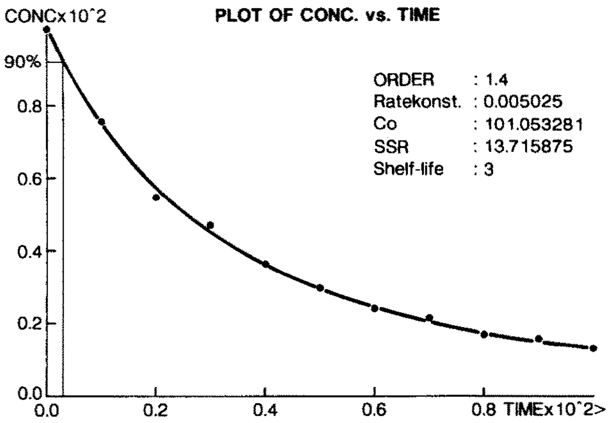


Fig. 2. Plot of concentration vs time for the best-fit order of a sample data set.

intercept by a call to the function wlinfit which performs a general weighted linear regression by calculation of the appropriate sums of weight, x values and y values.

From the slope and intercept of the linearized functions LINFIT calculates the best-fit rate constant and C_0 for the actual order. Finally, the SSR is calculated using the estimated parameters and input data.

In our opinion, graphic presentation of the calculated results is important for correct interpretation of the results. Following the calculations a screen showing the distribution of SSR as a function of reaction order is displayed (Fig. 1). The user selects a reaction order by moving an arrow. Hitting ENTER displays a plot of concentration vs time for the function with the selected reaction order along with the measured data (Fig. 2). This allows comparison of the calculated profiles for several different reaction orders.

In the program presented here the maximum order evaluated is 3.0, but increasing the value MAX_ORDER at the start of the program will also allow evaluation for higher orders. The program does not determine more than two significant figures for the orders, since this is considered sufficient for most purposes. However, increasing the number of significant figures is easily implemented by repeated variation of the order around the previously determined best-fit value.

Testing

The results from the program reported here were compared with those from general non-linear regression of the model given in Eqn 11. The data sets used were generated from the model in Eqn 11 with a random noise of $\pm 5\%$. Determination of the best-fit order was performed using a program for general non-linear regression (NLIN procedure in the SAS system (SAS Institute U.S., Cary, NC)), as well as the program reported here.

Performance

As can be seen from the results presented in Table 1, the differences between the general non-linear approach and our program were only due to the limited number of significant figures in the determina-

TABLE 1

Simulated parameters and values calculated by a general non-linear program and the reported program

	Simulation	SAS-NLIN (general)	SAS-NLIN (with two significant figures)	Current program
Order	0.0	0.0186020	0.0	0.0
Rate constant	1	0.9454194	1.0153646	1.013417
C_0	100	101.3616371	101.1079269	101.049490
SSR		41.997349	42.458619	42.498460
Order	0.2	0.2353683	0.2	0.2
Rate constant	0.5	0.4432631	0.5075352	0.507511
C_0	100	101.3242607	100.8431227	100.850951
SSR		12.722674	13.971595	13.972322
Order	1.0	1.0152811	1.0	1.0
Rate constant	0.05	0.0476753	0.0505267	0.050459
C_0	100	102.1985442	102.0708854	102.073571
SSR		4.324428	4.521181	4.530575
Order	1.4	1.4236461	1.4	1.4
Rate constant	0.005	0.0045938	0.0050380	0.005025
C_0	100	101.2173046	101.0226098	101.053281
SSR		13.419767	13.662899	13.715875
Order	2.0	1.99480981	2.0	2.0
Rate constant	0.0005	0.00050784	0.00049771	0.000495
C_0	100	96.30323938	96.32799903	96.357887
SSR		11.354356	11.361051	11.449400
Order	2.6	2.5925959	2.6	2.6
Rate constant	0.0001	0.0001069	0.0001040	0.000104
C_0	100	102.7566524	102.7637209	102.758985
SSR		1.522991	1.533153	1.540916
Order	3.0	2.91263522	2.9	2.9
Rate constant	0.00005	0.00006328	0.00006606	0.000066
C_0	100	99.06845894	99.06559382	99.064884
SSR		0.448382	0.462319	0.463782

tion of best-fit order. The time used to perform the necessary calculations even on a standard PC-XT (8088/4.77 MHz CPU) was 100 s for a dataset with 11 points.

Conclusion

The mathematical model described in this paper is versatile, since it covers the entire range of reaction orders. The linearization facilitates the computation and has been shown to produce results equivalent to a general non-linear approach. The program is easy to use and allows a number of possibilities for graphic presentation of the results.

Appendix

This appendix gives an outline of a complete program written in C, and the code necessary for the calculations.

```
/ * INCLUDE-FILES * /
#include ⟨stdio.h⟩
#include ⟨stdlib.h⟩
#include ⟨math.h⟩
#include ⟨float.h⟩
#include ⟨graph.h⟩
#include ⟨malloc.h⟩

/ * GLOBAL DEFINITIONS * /
#define MAX_ORDER      31     / *  Calculate orders from 0 - 3 with 1
                                   decimal accuracy                                * /
#define MAX_PKT        100    / *  Maximum number of measured points               * /
struct parset { double order,ratek,Co,SSR; };
struct linpars { double alfa,beta; };
/ * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * /
int datinp (xval,yval)
double * xval, * yval;
/ *   This function reads data from keyboard or file. The maximum number of points read is
      MAX_PKT. The time and concentration values are placed in two arrays starting at * xval and
      * yval respectively. The return value is the number of points read.
 * /
/ * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * /
double conc_calc(pars,tim)
struct parset * pars;
double tim;
/ *   This function calculates the concentration for a given time (tim) from the reaction order and
      parameters given in pars, using Eqn 11.                                               * /
/ * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * /
struct linpars * wlinreg(x,y,w,np)
double * x, * y, * w;
int np;
/ *   This function calculates the slope (beta) and intercept (alfa) from the np points in the arrays x,y
      and w (x value, y value and weight) using general weighted linear regression. Return value is a
      pointer to a structure containing alfa and beta.                                      * /
/ * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * /
struct parset * linfit(tval,cval,ordr,nopt)
double * tval, * cval,ordr;
int nopt;
/ *   This function is given in its entirety. It determines the best-fit parameters and squared sum of
      residuals for the given raw-data (tval = time, cval = concentration) and reaction-order (ordr). The
      number of points in the arrays are nopt. The calculated parameters are returned in a set
      (parset) * /
{     struct parset this_set;
      struct linpars these_pars;
```

```
    double weight [MAX_PKT],yval[MAX_PKT];
    int i;
    this_set.order = ordr;
    if (ordr = = 1)                                             / * Calculate the weights * /
      { for (i = 0,i < nopt;i + + )                                   / * for each point * /
        { weight[i] = pow(cval[i],2);
          if (cval[i] < = 0) yval[i] = log(DBL_MIN);
          else yval[i] = log(cval[i]);
        }
      }
    else
      { for (i = 0;i < nopt;i + + )
        { if (cval[i] < = 0)
            { weight[i] = pow(DBL_MIN,(2 * ordr))/pow((1-ordr),2);
              yval[i] = pow(DBL_MIN,(1-ordr));
            }
          else
            { weight[i] = pow(cval[i],(2 * ordr))/pow((1-ordr),2);
              yval[i] = pow(cval[i],(1-ordr));
            }
        }
      } / * if !(ordr = = 1) * /
/ * Perform the linear regression and transform the calculated parameters into rate-constant and
    Co                                                                                  * /
    these_pars =  * wlinreg(tval,yval,weight,nopt);
    if (ordr = = 1)
      { this_set.ratek = -these_pars.beta;
        this_set.Co = exp(these_pars.alfa);
      }
    else
      { this_set.ratek = these_pars.beta/(ordr-1);
        this_set.Co = pow(these_pars.alfa,1/(1-ordr));
      } / * if !(ordr = = 1) * /
    this_set.SSR = 0; / * Calculate the SSR * /
    for (i = 0;i < nopt;i + + )
      { this_set.SSR + = pow((cval[i]-
                              conc_calc(&this_set,tval[i])),2);
      }
    return(&this_set);
}
/ * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * /
void list_set(pos,heading,set)
int pos;
char * heading;
struct parset * set;
/ * This function is used to display the calculated parameters (set). The listing is started at position
    pos on the 3. line with a given heading, thus allowing appropriate placement according to the SSR
    function.                                                                           * /
```

```
/*************************************************/
void draw_func(param,timval,concval,nopt,video)
struct parset *param;
double *timval, *concval;
int nopt;
struct videoconfig *video;
/*  This function calculates and plots the concentration vs. time curve from the parameters given in
    param; along with the measured points given in timval and concval (nopt = the number of points).
    The type of graphics adapter is taken into account using the values of video.                  */
/*************************************************/
void graph_pres(ordset,tv,cv,n)
    struct parset *ordset;
    double *tv, *cv;
    int n;
/*  This function plots SSR vs reaction order given in ordset. The calculated values for the order
    indicated by a marker (an arrow, see Fig. 1) is displayed at an appropriate place on the screen.
    When enter is hit, draw_func is called to display the actual function for the selected order. Help is
    presented when F1 is hit. The raw data (n number of points of time (tv) and concentration (cv)) is
    included in the parameter list only to be transferred to draw_func.                             */
/*************************************************/
main ()
/*  Raw data are stored in timeval- and conc-arrays. The calculated best fit parameters are stored in
    the array of parset called ordarr                                                               */
{   double timeval [MAX_PKT],conc[MAX_PKT];
    double tmpdb;
    int no_of_points;
    int i,bf;
    struct parset ordarr[MAX_ORDER];
    no_of_points = datinp(timeval,conc);
    printf("\n\nCOMPUTING ");                               /* Calculate parameters */
    for (i = 0;i < MAX_ORDER;i++)                                  /* for each order */
      { tmpdb = i;
        ordarr[i] = *linfit(timeval,conc,tmpdb/10,no_of_points);
        printf(". ");
      };
    printf("FINISHED");
    if (_setvideomode(_MAXRESMODE))                        /* Enter graphics mode
                                                              if possible */
      { graph_pres(ordarr,timeval,conc,no_of_points);
        _setvideomode(_DEFAULTMODE);
      } /* end graphics OK */
    else                                                   /* graphics NOT OK,
                                                              print results */
      { tmpdb = ordarr[0].SSR;bf = 0;
        for (i = 1;i < MAX_ORDER;i++) /* Find set with the least SSR */
          if (tmpdb > ordarr[i].SSR) {tmpdb = ordarr[i].SSR;bf = i;};
        _clearscreen(_GCLEARSCREEN);
        _settextposition(1,30);
        printf("Touch any key to continue");
        for (i = 0;i < MAX_ORDER;i + + )
          { if (i = = bf)
                list_set(30,"  * * *  BEST FIT  * * * ",&ordarr[i]);
            else list_set(30,"                          ",&ordarr[i]);
            getch ( );
          }
      }       / * end else graphics NOT OK * /
    exit(0);
}
```

References

Connors, K.A., Amidon, G.L. and Valentino, J.S., *Chemical Stability of Pharmaceuticals,* Wiley, New York, 1986, p. 59.

Sande, S.A. and Karlsen, J., Curve fitting of dissolution data by personal computer. *Int. J. Pharm.*, 55 (1989) 193–198.

Thoma, K., *Artzneimittel Stabilität*, Frankfurt am Main, 1978, p. 286.