Power Series Method and Recurrences

Ian Smith

1 Power Series Method

The power series method (PSM) is a method to solve (by finding $u(t)$) a mathematical model described by a system of Ordinary Differential Equations.

$$\frac{du(t)}{dt} = \dot{u}(t) = ode(u(t))$$

to high accuracy (limited in practice by the platform precision), using arbitrary-order power series. In general, u is a composite variable, for example x, y, z coordinates in space. The PSM method itself treats these as independent solutions; there is no coupling within PSM, any coupling between variables happens entirely within the model, so a single variable, u, is used here for clarity.

2 The Algorithm

Evolution of a completely general time-varying quantity $u(t)$ is represented as a power series around a starting time t_0.

$$u = \sum_{k=0}^{\infty} U_k.(t - t_0)^k \tag{1}$$

Similarly $\dot{u}(t)$, the time derivative of $u(t)$ is represented as.

$$\dot{u} = \sum_{k=0}^{\infty} \dot{U}_k.(t - t_0)^k \tag{2}$$

Where the coefficients U_k and $\dot{U}_k$ (the dot always denotes time differentiation) are initially unknown (except for U_0, the value of $u(t_0)$), and are built up while incrementing the index k in an iterative procedure.

By explicitly differentiating (1) with respect to t, we obtain an alternative description of the derivative, this time in terms of U_k.

$$\dot{u} = \sum_{k=1}^{\infty} k.U_k.(t - t_0)^{k-1}$$

Rewriting with k starting from 0, to give equal powers of $(t - t_0)$ the same k index, turns the expression into a power series.

$$\dot{u} = \sum_{k=0}^{\infty} (k + 1).U_{k+1}.(t - t_0)^k \tag{3}$$

In a practical implementation, the sums are limited to a maximum k index, the order of the power series, n. U therefore has $n + 1$ coefficients (0 to n), and $\dot{U}$ has n coefficents (0 to $n - 1$).

Comparing (2) and (3) leads to the key identity of the method, used in two different ways.

$$\dot{U}_k = (k + 1).U_{k+1} \tag{4}$$

or, alternatively

$$U_{k+1} = \frac{\dot{U}_k}{k + 1} = \frac{ODE(U_{0..k})}{k + 1} \tag{5}$$

This identity is used in the form (5), looped over k, to build U via model-specific $ODE()$ equations, and is the core of the method. U_{k+1} is therefore derived from the output of the model, using *all* coefficients found so far (0 to k). The identity is also used, in the form (4), to replace time derivatives when deriving the chain rule recurrences below.

In summary, starting with initial value U_0, we build the power series one term at a time using (5), with $ODE()$ calling recurrence relations from below where needed, generating coefficients for k up to $n - 1$. The final k value in (5) produces U_n, and completes the ODE() phase for this iteration.

We then apply Horner's rule to U to calculate a new value $u(t_0 + h)$, where h is the time step).

$$U_0^{t_0+h} = \sum_{i=n}^{0} h.U_0^{t_0+h} + U_i \tag{6}$$

which becomes the initial value U_0 for the next iteration.

The power series method is described effectively as the combination of (5) and (6). It can be used "as is" to evolve linear coupled differential equations, but for nonlinear ones, we need to take steps to allow a proper calculation of $\dot{U}_k$. This process is highly model-specific, but typically involves one or more so-called recurrence relations, some examples of which are derived below.

3 Cauchy-based Recurrences

It is simplest to start with some basic algebra for power series.

3.1 Product

This is a straightforward Cauchy product of the two power series, each of $k+1$ coefficients (running from 0 to k).

$$P = U.V$$
$$P_k = \sum_{j=0}^{k} U_j.V_{k-j}$$

3.2 Quotient

Slightly more involved, we just need to write the quotient $Q = U/V$ as a product, then rearrange terms.

$$\begin{aligned} U &= Q.V \\ U_k &= \sum_{j=0}^{k} Q_j.V_{k-j} \\ &= \sum_{j=0}^{k-1} Q_j.V_{k-j} + Q_k V_0 \\ Q_0 &= U_0/V_0 \\ Q_k &= \frac{1}{V_0}(U_k - \sum_{j=0}^{k-1} Q_j.V_{k-j}) \end{aligned}$$

3.3 Reciprocal

As per quotient, with $U_0 = 1$, and $U_k = 0$ otherwise.

$$R_0 = 1/V_0$$
$$R_k = -\frac{1}{V_0} \sum_{j=0}^{k-1} Q_j.V_{k-j}$$

3.4 Square

Very similar to product

$$S = U.U$$

$$S_k = \sum_{j=0}^{k} U_j.U_{k-j}$$

Note that it is possible to exploit the symmetry of this solution, and only calculate "half" of the coefficients. There are two cases to consider; k odd and k even. Bear in mind the number of coefficients is $k+1$!

$$S_k^{odd} = 2 \sum_{j=1}^{(k-1)/2} U_j.U_{k-j}$$

$$S_k^{even} = 2 \sum_{j=1}^{(k-2)/2} U_j.U_{k-j} - 2U_{k/2}^2$$

3.5 Square root

This is another re-arranged product, but relies on a "seed value" for $k = 0$, from a mathematical function call.

$$U = R.R$$

$$U_k = \sum_{j=0}^{k} R_j.R_{k-j}$$

$$= \sum_{j=1}^{k-1} R_j.R_{k-j} + 2R_0R_k$$

$$R_0 = sqrt(U_0)$$

$$R_k = \frac{1}{2R_0}(U_k - \sum_{j=1}^{k-1} R_j.R_{k-j})$$

As for square, it is possible to exploit the symmetry of this solution, and only calculate "half" of the coefficients.

$$R_k^{odd} = \frac{1}{2R_0}(U_k - 2 \sum_{j=1}^{(k-1)/2} R_j.R_{k-j})$$

$$R_k^{even} = \frac{1}{2R_0}(U_k - 2 \sum_{j=1}^{(k-2)/2} R_j.R_{k-j} - R_{k/2}^2)$$

4 Chain Rule Recurrences

To deal with functions of power series, we need to handle composition, that is things like $f(u(t))$. To do this, apply the chain rule to the time derivative of a composed function, $\frac{d}{dt}f(u(t))$. This creates another Cauchy product, each series this time having k coefficients (indices running from 0 to $k-1$).

We then use the identity (4) from earlier to replace the time derivatives.

$$\dot{F} = \frac{df}{dt} = \frac{df}{du}\frac{du}{dt} = dFdU.\dot{U}$$
$$\dot{F}_{k-1} = \sum_{j=0}^{k-1} dFdU_j.\dot{U}_{k-1-j}$$
$$kF_k = \sum_{j=0}^{k-1} dFdU_j.(k-j)U_{k-j}$$

These recurrences do not cater for the case $k = 0$; this requires seeding externally by a mathematical library call.

4.1 Forward

$$F_0 = f(U_0)$$
$$F_k = \frac{1}{k}\sum_{j=0}^{k-1} dFdU_j.(k-j)U_{k-j}$$

4.2 Reverse

$$U_0 = f^{-1}(F_0)$$
$$U_k = \frac{1}{dFdU_0}(F_k - \frac{1}{k}\sum_{j=1}^{k-1} dFdU_j.(k-j)U_{k-j})$$

5 $exp(), log()$

This is a good introduction to the use of forward and reverse recurrences. This case is special because $dEXPdU = EXP$ ($dFdU = F$).

5.1 $exp()$ - forward

$$EXP_0 = exp(U_0)$$
$$EXP_k = \frac{1}{k}\sum_{j=0}^{k-1} EXP_j.(k-j)U_{k-j}$$

5.2 $ln()$ - reverse

$$U_0 = ln(EXP_0)$$
$$U_k = \frac{1}{EXP_0}(EXP_k - \frac{1}{k}\sum_{j=1}^{k-1} EXP_j.(k-j)U_{k-j})$$

6 trigonometric & hyperbolic functions

Like *exp*(), these are all *forward* recurrences. They must be built as a pair, as each depends on the other.

6.1 $sin(), cos()$

$$dSINdU = COS$$
$$SIN_0 = sin(U_0)$$
$$SIN_k = \frac{1}{k}\sum_{j=0}^{k-1} COS_j.(k-j)U_{k-j}$$

$$dCOSdU = -SIN$$
$$COS_0 = cos(U_0)$$
$$COS_k = \frac{-1}{k}\sum_{j=0}^{k-1} SIN_j.(k-j)U_{k-j}$$

6.2 $sinh(), cosh()$

$$dSINHdU = COSH$$
$$SINH_0 = sinh(U_0)$$
$$SINH_k = \frac{1}{k}\sum_{j=0}^{k-1} COSH_j.(k-j)U_{k-j}$$

$$dCOSHdU = SINH$$
$$COSH_0 = cosh(U_0)$$
$$COSH_k = \frac{1}{k}\sum_{j=0}^{k-1} SINH_j.(k-j)U_{k-j}$$

6.3 $tan(), sec^2()$

$$dTANdU = SEC^2$$
$$TAN_0 = tan(U_0)$$
$$TAN_k = \frac{1}{k}\sum_{j=0}^{k-1} SEC_j^2.(k-j)U_{k-j}$$

$$dSEC^2dU = 2TAN$$
$$SEC_0^2 = sec^2(U_0)$$
$$SEC_k^2 = \frac{2}{k}\sum_{j=0}^{k-1} TAN_j.(k-j)TAN_{k-j}$$

6.4 $tanh(), sech^2()$

$$dTANHdU = SECH^2$$
$$TANH_0 = tanh(U_0)$$
$$TANH_k = \frac{1}{k}\sum_{j=0}^{k-1} SECH_j^2.(k-j)U_{k-j}$$

$$dSECH^2dU = -2TANH$$
$$SECH_0^2 = sech^2(U_0)$$
$$SECH_k^2 = \frac{-2}{k}\sum_{j=0}^{k-1} TANH_j.(k-j)TANH_{k-j}$$

7 Inverse trigonometric & hyperbolic functions

Like $ln()$, these are all *reverse* recurrences. Each needs to be paired with the same *forward* recurrence as in the non-inverse case.

7.1 $asin()$

$$U_0 = asin(SIN_0)$$

$$U_k = \frac{1}{COS_0}(SIN_k - \frac{1}{k}\sum_{j=1}^{k-1} COS_j.(k-j)U_{k-j})$$

7.2 $asinh()$

$$U_0 = asinh(SINH_0)$$

$$U_k = \frac{1}{COSH_0}(SINH_k - \frac{1}{k}\sum_{j=1}^{k-1} COSH_j.(k-j)U_{k-j})$$

7.3 $acos()$

$$U_0 = acos(COS_0)$$

$$U_k = \frac{-1}{SIN_0}(COS_k - \frac{1}{k}\sum_{j=1}^{k-1} -SIN_j.(k-j)U_{k-j})$$

7.4 $acosh()$

$$U_0 = acosh(COSH_0)$$

$$U_k = \frac{1}{SINH_0}(COSH_k - \frac{1}{k}\sum_{j=1}^{k-1} SINH_j.(k-j)U_{k-j})$$

7.5 $atan()$

$$U_0 = atan(TAN_0)$$

$$U_k = \frac{1}{SEC_0^2}(TAN_k - \frac{1}{k}\sum_{j=1}^{k-1} -SEC_j^2.(k-j)U_{k-j})$$

7.6 $atanh()$

$$U_0 = atanh(TANH_0)$$

$$U_k = \frac{1}{SECH_0^2}(TANH_k - \frac{1}{k}\sum_{j=1}^{k-1} SECH_j^2.(k-j)U_{k-j})$$

8 Power Recurrence

Creates two power series, then combines them into a third loop form (alongside the Cauchy and chain rule forms).

$$P = U^a$$

Use the chain rule to create a product, multiply both sides by U, then replace the time derivatives.

$$\begin{aligned}
\frac{dp}{dt} &= \frac{dp}{du}.\frac{du}{dt} \\
\dot{P} &= a.U^{a-1}.\dot{U} \\
U.\dot{P} &= a.P.\dot{U} \\
\sum_{j=0}^{k-1} U_j.(k-j)P_{k-j} &= a\sum_{j=0}^{k-1} P_j.(k-j)U_{k-j} \\
U_0.k.P_k + \sum_{j=1}^{k-1} U_j.(k-j)P_{k-j} &= a\sum_{j=0}^{k-1} P_j.(k-j)U_{k-j} \\
P_k &= \frac{1}{kU_0}(a\sum_{j=0}^{k-1} P_j.(k-j)U_{k-j} - \sum_{j=1}^{k-1} U_j.(k-j)P_{k-j})
\end{aligned}$$

The index ranges in the second sum now match, so we can flip the indexing for U and P.

$$P_k = \frac{1}{kU_0}(a\sum_{j=0}^{k-1} P_j.(k-j)U_{k-j} - \sum_{j=1}^{k-1} U_{k-j}.jP_j)$$

Since the additional $j = 0$ term is $U_k.0.P_0$, we can match the summation indices up with the first sum.

$$P_k = \frac{1}{kU_0}(a\sum_{j=0}^{k-1} P_j.(k-j)U_{k-j} - \sum_{j=0}^{k-1} U_{k-j}.jP_j)$$

$$\begin{aligned}
P_0 &= pow(U_0, a) \\
P_k &= \frac{1}{kU_0}\sum_{j=0}^{k-1}(a(k-j)-j).P_j.U_{k-j}
\end{aligned}$$

THE END