Parameters related to explicit methods

For transient simulation of a circuit with explicit compound elements (XCEs), SEQUEL employs explicit methods[1] (see Chapter 4 of Part-1). The following parameters apply in that case.

* `forward_euler`: (`yes`/`no`) for Forward Euler method with constant time step.
* `RK4`: (`yes`/`no`) for Runge-Kutta order-4 method with constant time step.
* `modified_euler`: (`yes`/`no`) for Modified (Improved) Euler method (see [1], for example) with constant time step.
* `Heun`: (`yes`/`no`) for Heun method (see [1], for example) with constant time step.

  When a constant step method (Forward Euler, RK4, Modified Euler, Heun) is used, a few smaller time steps may be taken to account for corners in input waveforms, for example.
* `RKF45`: (`yes`/`no`) for Runge-Kutta-Fehlberg 4/5 method (auto time step)
* `BS23`: (`yes`/`no`) for Bogacki-Shampine 2/3 method [2] (auto time step)
* `delt_const_x`: (real number) serves as the constant time step for Forward Euler, RK4, Modified Euler, and Heun methods, and as the first time step for methods with auto time steps.
* `delt_min_x`: (real number) smallest time step allowed (default: $0.0002 \times$ `delt_const_x`)
* `delt_max_x`: (real number) largest time step allowed (default: $10 \times$ `delt_const_x`)

Next, we list parameters related to the RKF45 and BS23 methods. These methods use auto time stepping. At each time point, a multiplier k is computed from the tolerance and an estimate of the local truncation error (see Sec. 4.4 of Part-1) to obtain the next time step as

$$\Delta t^{\text{new}} = k \times \Delta t^{\text{old}}. \tag{1}$$

The parameters are

* `rkf45_tolr`: (real number) tolerance value for the RKF45 method (default: 10^{-8})
* `rkf45_fctr_min`: (real number) lower limit on multiplier k (Eq. 1) in the RKF45 method. (default: 0.8)
* `rkf45_fctr_max`: (real number) upper limit on multiplier k (Eq. 1) in the RKF45 method. (default: 1.1)
* `bs23_tolr`: (real number) tolerance value for the BS23 method (default: 10^{-8})
* `bs23_fctr_min`: (real number) lower limit on multiplier k (Eq. 1) in the BS23 method. (default: 0.8)

[1] With explicit elements, only transient simulation is allowed; DC, start-up, AC, SSW are not allowed.

* `bs23_fctr_max`: (real number) upper limit on multiplier k (Eq. 1) in the BS23 method. (default: 1.1)

As we have seen in Sec. 4.6 of Part-1, algebraic loops create a difficulty for explicit methods. If the user's system has algebraic loops, the algebraic equations must be solved separately. As an example, consider the system shown in Fig. 1 which has two algebraic loops (the loops involving multipliers k_1 and k_2).

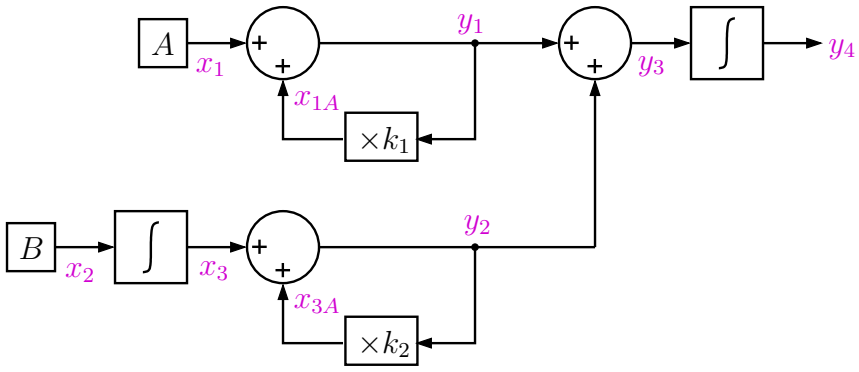

Figure 1: A system with algebraic loops. A and B are constants.

In such a case, SEQUEL would first update the variables associated with a time derivative, in this case, x_3 and y_4. For example, with the Forward Euler method, we have

$$y_4^{n+1} = y_4^n + hy_3^n, \tag{2}$$

$$x_3^{n+1} = x_3^n + hx_2^n, \tag{3}$$

where $h = t_{n+1} - t_n$ is the time step.

Having obtained y_4^{n+1} and x_3^{n+1}, the other variables are updated by solving the algebraic equations governing those variables. In the above example, all elements are linear, so it is a simple matter of solving a linear system of equations. If there are nonlinear elements in the system, the Newton-Raphson (NR) method (see Chapter 3 of Part-1) is used to solve the resulting equations.

Apart from algebraic loops, there is another situation in which a linear or nonlinear system of equations needs to be solved, and that is the presence of electrical-type elements which are implemented as XCEs. In that case, SEQUEL internally adds the required KCL and KVL equations.

The overall set of equations is divided into two sub-sets: (a) ODEs, (b) algebraic equations. An algebraic equation may arise from an algebraic loop, it may be a KCL/KVL equation, or it may be an element behaviour equation. The two sub-sets are treated separately – the ODEs with an explicit method (specified by the user) and the algebraic equations with a linear or nonlinear solver.

The following parameters are relevant in the above context.

* `x_eval_serial`: (`yes`/`no`) is used to indicate whether the elements should be evaluated in a serial fashion (see Sec. 4.6 in Part-1). It should be set to `no` when there are algebraic loops or electrical-type XCEs in the system.

Default: If there are electrical-type XCEs, `x_eval_serial` is set to `no` by default; else, it is set to `yes`.

* `x_itmax_newton`: (integer) maximum number of NR iterations (default: 500).

* `x_dmp`: (`yes`/`no`) decides whether damping (see Eq. 3.19 in Part-1) should be used (default: `no`)

* `x_dmp_k`: (real number) damping factor k where $0 < k < 1$ (see Eq. 3.19 in Part-1, default: 0.2). Not relevant when `x_dmp` is set to `no`.

* `x_dmp_newt_max`: (integer) number of NR iterations for which damping is applied (default: 50). Not relevant when `x_dmp` is set to `no`.

* `x_chk_rhs2`: (`yes`/`no`) decides whether 2-norm (see Eq. 3.9 in Part-1) should be used to check for convergence of NR iterations. (default: `yes`)

* `x_norm_2`: (real number) tolerance value for the 2-norm (default: 10^{-10}). Not relevant when `x_chk_rhs2` is `no`.

* `x_write_rhs2`: (`yes`/`no`) decides whether the 2-norm should be written to the console (for each NR iteration). Default: `no`.

* `x_chk_spice`: (`yes`/`no`) decides whether the SPICE convergence criteria (see Sec. 3.3 in Part-1) should be used to check for convergence of NR iterations. (default: `no`)

* `x_norm_spice_rel`: (real number) k_{rel} in Eq. 3.10 of Part-1 (default: 10^{-3}).

* `x_norm_spice_nodev`: (real number) τ_{abs} for node voltages (see Eq. 3.10 of Part-1, default in Volts: 10^{-6}).

* `x_norm_spice_cur`: (real number) τ_{abs} for currents (default in Amps: 10^{-12}).

* `x_norm_spice_xaux`: (real number) τ_{abs} for XBE auxiliary variables (default: 10^{-4}).

  In general, it is difficult to set `x_norm_spice_xaux` in a meaningful manner because it corresponds to variables of different kinds. For example, an auxiliary variable in an XBE may be a speed or force or current. It is made available to the user mainly for the sake of completeness.

* `x_write_spice`: (`yes`/`no`) decides whether information about SPICE convergence parameters should be written to the console (for each NR iteration). Default: `no`.

As we have seen in Chapter 3 of Part-1, convergence of the NR process depends on the initial guess. Convergence at the very first time point in transient simulation is more difficult because we may not have a good initial guess to start the NR process. For subsequent time points, the solution obtained at the previous time point generally serves as an excellent initial guess, and convergence is easier. For this reason, NR parameters for the first solution are made available separately, as given below.

* `x_itmax_newton_first`: (integer) maximum number of NR iterations for the first solution (default: 500).

* `x_dmp_first`: (`yes`/`no`) decides whether damping should be used for the first solution (default: `no`)

* `x_dmp_k_first`: (real number) damping factor k ($0 < k < 1$) for the first solution (default: 0.1). Not relevant when `x_dmp_first` is set to `no`.

* `x_dmp_newt_max_first`: (integer) number of NR iterations for which damping is applied for the first solution (default: 50). Not relevant when `x_dmp_first` is set to `no`.

References

1. M. B. Patil, V. Ramanarayanan, and V. T. Ranganathan, *Simulation of Power Electronic Circuits*, Narosa, New Delhi, 2009.

2. `https://en.wikipedia.org/wiki/Bogacki%E2%80%93Shampine_method`