odepkg: Present and Future

J. Corno
Outline

1. Release 0.8.5
2. SoCiS 2013
3. SoCiS 2014
4. TODOs and Discussion
Release 0.8.5

- Released on 19 May 2015
- Octave 4.0 compatibility (Tatsuro Matsuoka)
**New solvers**: Geometric integrators for Hamiltonian Systems

- `odeSE` (Symplectic Euler)
- `odeSV` (Stormer Verlet)
- `odeVV` (Velocity Verlet)
- `odeSPVI` (SPectral Variational Integrator)
- `odeRATTLE` (RATTLE algorithm)

**New options handling**

- Levenshtein
- Fuzzy compare

**New structure**

- Steppers
- Integrate functions
The new organization tries to subdivide the code according to the most important operations.

**Optimization** of the bottlenecks and **extension** of the code should be easier.
A **stepper** executes just one integration step

\[
[x_{\text{next}}, \text{err}] = \text{stepper}(f, x, t, dt)
\]

The estimation of the error is used to determine the optimal $dt$ in adaptive integrators.
Integrate Functions

Given the stepper, an **integrate function** executes the integration algorithm on more steps

- `integrate_const (stepper, f, t, x0, dt, opts)`
- `integrate_n_steps (stepper, f, t0, x0, dt, n, opts)`
- `integrate_adaptive (stepper, p, f, t, x0, opts)`
All Together: ode45

```matlab
switch integrate_func
    case 'const'
        solution = integrate_const (@runge_kutta_45_dorpri,
                                  fun, tvec, x0, dt, opts);
    case 'n_steps'
        solution = integrate_n_steps (  
                                       @runge_kutta_45_dorpri, fun, t0, x0, dt, n,  
                                       opts);
    case 'adaptive'
        solution = integrate_adaptive ( 
                                        @runge_kutta_45_dorpri, order, fun, tvec, x0,  
                                        opts);
endswitch
```
New structure for all the explicit solvers

odeset and odeget Matlab compatible

Implementation of the missing options (MaxStep, NormControl, etc...)

Tests added and bugfixes

First Same As Last (when possible)

Dense output
Dense Output

Provide the solution at a given time \( s \in [t, t + dt] \) with the same order of accuracy as the solutions computed at the internal time points by using suitable interpolation methods.

- \( \text{x\_out} = \text{linear\_interpolation} \ (t, \ x, \ t\_out) \)
- \( \text{x\_out} = \text{quadratic\_interpolation} \ (t, \ x, \ \text{der}, \ t\_out) \)
- \( \text{x\_out} = \text{hermite\_cubic\_interpolation} \ (t, \ x, \ \text{der}, \ t\_out) \)
- \( \text{x\_out} = \text{hermite\_quartic\_interpolation} \ (t, \ x, \ \text{der}, \ t\_out) \)
- \( \text{x\_out} = \text{dorpri\_interpolation} \ (t, \ x, \ k, \ t\_out) \)
- \( \text{x\_out} = \text{hermite\_quintic\_interpolation} \ (t, \ x, \ \text{der}, \ t\_out) \)
Dense Output

- 1\textsuperscript{st} order approximation with no function evaluation
- 2\textsuperscript{nd} order approximation may require the evaluation of the function at the current time. Avoided if the stepper already returns that value
- The only 3\textsuperscript{rd} explicit order solver implemented is \texttt{ode23}. The 3\textsuperscript{rd} order approximation exploits the Runge-Kutta $k$ values to avoid further function evaluations.
- If \texttt{ode45} is used without local extrapolation then \texttt{dorpri_interpolation} gives 4\textsuperscript{th} order approximation without any additional function evaluation
For **ode45** with local extrapolation, Shampine proposes 4th order approximation at the middle point and to use quartic interpolation. The quintic interpolation requires an additional function evaluation without (according to Shampine) a significant improvement.

For the higher order solvers (**ode78**), a suitable interpolator has not yet been implemented.

Further optimization can be performed:
- If more than one solution is requested in \( s \in [t, t + dt] \)
- For specific solvers
TODOs and Discussion

- Clean-up and release with the new structure (0.9.0 ?)
- **Move to core** the most used solvers
  - `ode45`
  - `ode23`
  - `ode23s`
  - `ode15s` (to be implemented!)
  - `bvp4c` (and other BVP?)
  - `odeset` & `odeget`

**N.B.** To move the main solvers to core it is necessary to move also
- The utilities for the options (`levenshtein`, ...)
- The three **integrate functions**
- The **steppers** corresponding to the solvers
- ...
Todos and Discussion

Questions for discussion:

- **inputParser** for `odeset/odeget`
- When to move? Before or after new release?
- What happens to `daspk`, `dassl`, ...
  - Remove
  - Keep and change interface according to `odeset/odeget`
  - New wrapper to mimic `ode15s`

Longer term TODO:

- Implement a MATLAB compatible version of `deval`
- Better handling of the options (avoid so much code repetition)