https://docs.scipy.org/doc/scipy/reference/integrate.html#solving-initial-value-problems-for-ode-systems из справочного руководства SciPy.org

Интеграция и ODE (scipy.integrate) ¶

Интегрирующие функции, заданные функциональным объектом

quad(func, a, b [, args, full_output,…])	Вычислить определенный интеграл.
quad_vec(f, a, b [, epsabs, epsrel, норма,…])	Адаптивное интегрирование вектор-функции.
dblquad(func, a, b, gfun, hfun [, args,…])	Вычислить двойной интеграл.
tplquad(func, a, b, gfun, hfun, qfun, rfun)	Вычислить тройной (определенный) интеграл.
nquad(func, range [, args, opts, full_output])	Интеграция по нескольким переменным.
fixed_quad(func, a, b [, args, n])	Вычислить определенный интеграл, используя квадратуру Гаусса фиксированного порядка.
quadrature(func, a, b [, args, tol, rtol,…])	Вычислить определенный интеграл, используя гауссовскую квадратуру с фиксированным допуском.
romberg(функция, a, b [, args, tol, rtol,…])	Интеграция Ромберга вызываемой функции или метода.
quad_explain([выход])	Вывести дополнительную информацию о параметрах integrate.quad () и возвратах.
newton_cotes(pH [, равно])	Возвращаемые веса и коэффициент ошибок для интеграции Ньютона-Кота.
IntegrationWarning	Предупреждение о проблемах при интеграции.

Интегрирующие функции при фиксированных выборках

trapz(y [, x, dx, ось])	Интегрируйте вдоль заданной оси, используя сложное трапециевидное правило.
cumtrapz(y [, x, dx, ось, начальная])	Кумулятивно интегрировать y (x), используя составное правило трапеции.
simps(y [, x, dx, ось, даже])	Интегрируем y (x), используя выборки вдоль заданной оси и составное правило Симпсона.
romb(y [, dx, ось, показать])	Интегрирование Ромберга с использованием примеров функции.

Смотрите также

scipy.special для ортогональных многочленов (специальных) для гауссовых квадратурных корней и весов для других весовых коэффициентов и областей.

Решение начальных задач для систем ODE

Решатели реализованы в виде отдельных классов, которые можно использовать напрямую (низкоуровневое использование) или с помощью вспомогательной функции.

solve_ivp(забавно, t_span, y0 [, method, t_eval,…])	Решить проблему начальных значений для системы ODE.
RK23(забавно, t0, y0, t_bound [, max_step, rtol,…])	Явный метод Рунге-Кутты порядка 3 (2).
RK45(забавно, t0, y0, t_bound [, max_step, rtol,…])	Явный метод Рунге-Кутты порядка 5 (4).
DOP853(забавно, t0, y0, t_bound [, max_step,…])	Явный метод Рунге-Кутты порядка 8.
Radau(забавно, t0, y0, t_bound [, max_step,…])	Неявный метод Рунге-Кутты семейства Радау IIA порядка 5.
BDF(забавно, t0, y0, t_bound [, max_step, rtol,…])	Неявный метод, основанный на формулах обратной дифференциации.
LSODA(забавно, t0, y0, t_bound [, first_step,…])	Метод Адамса / BDF с автоматическим определением жесткости и переключением.
OdeSolver(забавно, t0, y0, t_bound, векторизация)	Базовый класс для ODE решателей.
DenseOutput(t_old, t)	Базовый класс для локального интерполяции на шаге, сделанный решателем ODE.
OdeSolution(т.с., интерполанты)	Непрерывное решение ODE.

Старый API

Это процедуры, разработанные ранее для sciру. Они заключают в себе старые решатели, реализованные на Фортране (в основном ODEPACK). Хотя интерфейс к ним не особенно удобен и некоторые функции отсутствуют по сравнению с новым API, сами решатели имеют хорошее качество и работают быстро, как скомпилированный код на Фортране. В некоторых случаях, возможно, стоит использовать этот старый API.

odeint(func, y0, t [, args, Dfun, col_deriv,…])	Интегрируем систему обыкновенных дифференциальных уравнений.
ode(f [, jac])	Универсальный класс интерфейса для числовых интеграторов.
complex_ode(f [, jac])	Оболочка из оды для сложных систем.

Решение краевых задач для систем ОДУ

solve_bvp(веселье, до н.э, x, y [, p, S, fun_jac,…])