$$y'' + y = \sin(t), \quad y(0) = 1, y'(0) = 0$$

の数値近似解を求めたい. $f = y, g = y'$ と置く.

$$\frac{d}{dt}\begin{pmatrix} f \\ g \end{pmatrix} = \begin{pmatrix} g \\ -f - \sin(t) \end{pmatrix}$$

GSL の Runge-Kutta 法

コンパイルと実行.

```
gcc rk-0517c.c -lgsl -lm
./a.out
```

C 言語のソースコード rk-0517c.c [1]

```
#include <stdio.h>
#include <math.h>
#include <gsl/gsl_errno.h>
#include <gsl/gsl_matrix.h>
#include <gsl/gsl_odeiv.h>
int func (double x, const double y[],
  double f[], void *params) {
  f[0] = y[1];
  f[1] = -y[0]-sin(x/3.0);
  return GSL_SUCCESS;
}
int main (void)
{/* rank 2 の方程式を解くステップ関数を生成 */
  const gsl_odeiv_step_type *T = gsl_odeiv_step_rk4;
  gsl_odeiv_step *s = gsl_odeiv_step_alloc(T, 2);
/* 各ステップにおいて解の値 y_i(t) に対する誤差を絶対誤差 1e-6
相対誤差 1e-5 の範囲内に抑えるようなステップ調整法を生成 */
  gsl_odeiv_control *c
  = gsl_odeiv_control_y_new(1e-6, 1e-5);

/* rank 2 の方程式のための求解関数の生成 */
  gsl_odeiv_evolve *e = gsl_odeiv_evolve_alloc(2);
  gsl_odeiv_system sys = {func, NULL, 2, NULL};
  /* x : start, x1 : goal */
  double x, x1 = 10.0;
  double h = 1e-6; /* ステップ幅の初期値 */
  double y[2];
  x = 0.0;
  y[0]=1.0; y[1]=0.0;  /* y の初期値 */
  while (x < x1) {
    int status
     =gsl_odeiv_evolve_apply(e, c, s,&sys, &x, x1, &h, y);
    if (status != GSL_SUCCESS) break;
    printf("x=%.5e, y[0]=%.5e, h=%lg\n", x, y[0],h);
  }
  gsl_odeiv_evolve_free(e);
  gsl_odeiv_control_free(c);
  gsl_odeiv_step_free(s);
  return 0;
}
```

[1] 講義のページ http://www.math.kobe-u.ac.jp/HOME/taka/2022/c1/ref.html の “プログラム等” からダウンロードできます.