数値計算

常微分方程式の数値解法

常微分方程式の数値解法

$$\frac{d}{dx}y(x) = f(x,y), \quad y(x_0) = y_0$$

- オイラー法
- 修正オイラー法
- ルンゲ・クッタ法

常微分方程式の数値解法の基礎

$$\frac{d}{dx}y(x) = f(x, y(x))$$

$$\int_a^{a+h} \frac{d}{dx}y(x)dx = \int_a^{a+h} f(x, y(x))dx$$

$$y(a+h) - y(a) = \int_a^{a+h} f(x, y(x))dx$$

$$y(a+h) = y(a) + \int_a^{a+h} f(x, y(x))dx$$

多階常微分方程式

$$\frac{d^n y}{dx^n} = f(x, y, y^{(1)}, y^{(2)}, \cdots, y^{(n-1)}) \qquad y^{(m)} = \frac{d^m y}{dx^m}$$

$$z_0 \equiv y(x), \ z_k \equiv \frac{d^k y}{dx^k}, \ (k = 1, 2, 3, \cdots, n-1)$$

$$\begin{cases} \frac{d}{dx}z_0 = z_1 \\ \frac{d}{dx}z_1 = z_2 \\ \frac{d}{dx}z_2 = z_3 \\ \vdots \\ \frac{d}{dx}z_{n-1} = f(x, z_0, z_1, z_2, \cdots, z_{n-1}) \end{cases}$$

$$\vec{y}(x) = \begin{pmatrix} z_0 \\ z_1 \\ z_2 \\ \vdots \\ z_{n-1} \end{pmatrix} \qquad \frac{d}{dx}\vec{y}(x) = \vec{F}(x, \vec{y}(x))$$

オイラー法

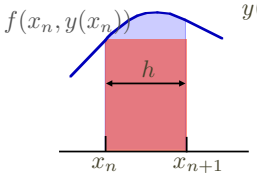
$$\frac{d}{dx}y(x) = f(x, y(x))$$

$$y(a+h) = y(a) + \int_a^{a+h} f(x, y(x))dx$$

$$x_{n+1} = x_n + h$$

$$y(x_{n+1}) = y(x_n) + \int_{x_n}^{x_{n+1}} f(x, y(x))dx$$

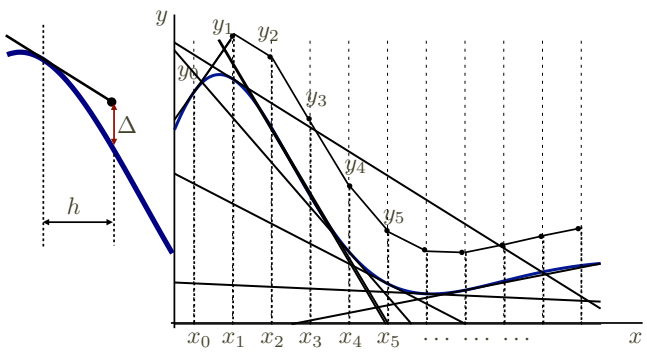
$$= y(x_n) + hf(x_n, y(x_n))$$



$$\begin{aligned} x_{n+1} &= x_n + h \\ y_n &= y(x_n) \\ y_{n+1} &= y_n + hf(x_n, y_n) \end{aligned}$$

オイラー法

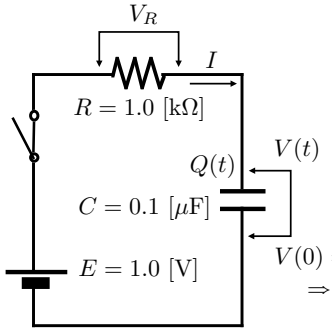
$$y_{n+1} = y_n + hf(x_n, y_n) + O(h^2)$$



オイラー法のプログラム

```
#include <stdio.h> /* 必要ならば math.h も */
double dydx(double x, double y);
void output(double x, double y);
int main(void) {
  int i, n = 100; /* 繰り返し回数 */
  double x = 0.0, y = 0.0; /* 初期値 */
  double dx = 0.01;        /* 刻み幅 */
  double dy;
  output(x, y);
  for (i=0; i<n; i++) {
    dy = dydx(x, y)*dx;
    x = x + dx;
    y = y + dy;
    output(x, y);
  }
  return 0;
}
```

例題1



$$V_R = IR = R\frac{d}{dt}Q(t)$$
$$Q(t) = CV(t)$$
$$V_R + V(t) = E$$
$$RC\frac{d}{dt}V(t) + V(t) = E$$

$$V(0) = 0 \Rightarrow V(t) = E\left(1 + \exp\left[-\frac{t}{RC}\right]\right)$$

$$RC = 1.0\ [\mathrm{k\Omega}] \times 0.1\ [\mu\mathrm{F}] = 0.1[\mathrm{ms}]$$

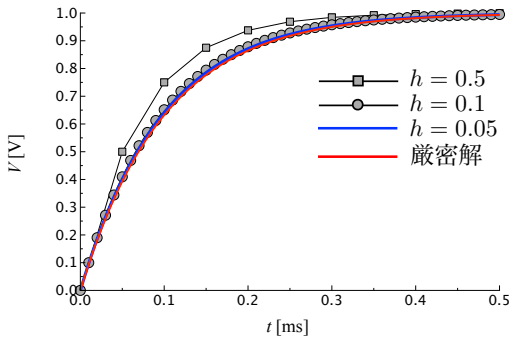
例題1

$$\frac{d}{dt}V(t) = -\frac{1}{RC}V(t) + \frac{E}{RC} \qquad E = 1.0\ [\mathrm{V}]$$
$$RC = \tau = 0.1\ [\mathrm{ms}]$$

$$\left.\begin{aligned} x &= \frac{t}{RC} = \frac{t}{\tau} \\ y &= \frac{V(\tau x)}{E} \end{aligned}\right\} \Rightarrow \frac{d}{dx}y(x) = -y(x) + 1$$

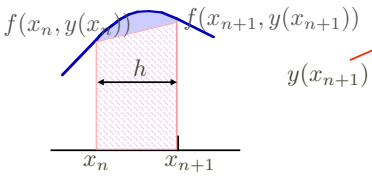
```
double dydx(double x, double y) {
  return (1.0 - y);
}
void output(double x, double y) {
  double e0 = 1.0; /* E=1.0[V] */
  double t0 = 0.1; /* tau=0.1[ms] */
  printf ("%f %f\n", x*t0, y*e0);
}
```

例題1(オイラー法)



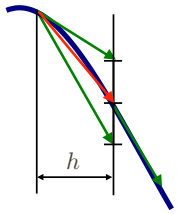
修正オイラー法

$$x_{n+1} = x_n + h$$
$$y(x_{n+1}) = y(x_n) + \int_{x_n}^{x_{n+1}} f(x, y(x))dx$$
$$= y(x_n) + h\frac{f(x_n, y(x_n)) + f(x_{n+1}, y(x_{n+1}))}{2}$$



$$y(x_{n+1}) = y(x_n) + hf(x_n, y(x_n))$$

修正オイラー法



$$x_{n+1} = x_n + h$$
$$y_n = y(x_n)$$

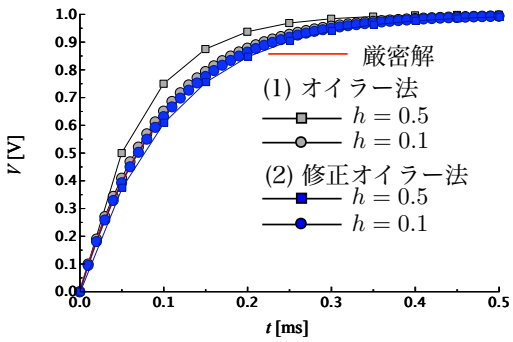
$$Y_n^* = y_n + hf(x_n, y_n)$$

$$y_{n+1} = y_n + \frac{h}{2}\left[f(x_n, y_n) + f(x_{n+1}, Y_n^*)\right]$$

修正オイラー法のプログラム

```
#include <stdio.h>    /* 必要ならば math.h も */
double dydx(double x, double y);
void output(double x, double y);
int main(void) {
  int i, n = 100;/* 繰り返し回数 */
  double x = 0.0, y = 0.0;  /* 初期値 */
  double dx = 0.01;  /* 刻み幅 */
  double dy1, dy2;
  output(x, y);
  for (i=0; i<n; i++) {
    dy1 = dydx(x, y)*dx;
    dy2 = dydx(x + dx, y + dy1)*dx;
    x = x + dx;
    y = y + (dy1 + dy2)/2.0;
    output(x, y);
  }
  return 0;
}
```

例題1(修正オイラー法)

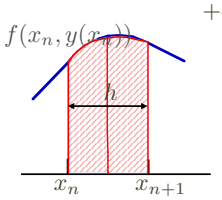


ルンゲ・クッタ法

$$x_{n+1} = x_n + h$$

$$y(x_{n+1}) = y(x_n) + \int_{x_n}^{x_{n+1}} f(x, y(x))dx$$

$$y(x_{n+1}) = y(x_n) + \frac{h}{2}\frac{1}{3}[f(x_n, y(x_n)) + 4f(x_n + \tfrac{h}{2}, y(x_n + \tfrac{h}{2})) + f(x_{n+1}, y(x_{n+1}))]$$



ルンゲ・クッタ法

$$x_{n+1} = x_n + h$$

$$y_n = y(x_n)$$

$$k_1 = f(x_n, y_n)$$

$$k_2 = f(x_n + \frac{h}{2}, y_n + \frac{h}{2}k_1)$$

$$k_3 = f(x_n + \frac{h}{2}, y_n + \frac{h}{2}k_2)$$

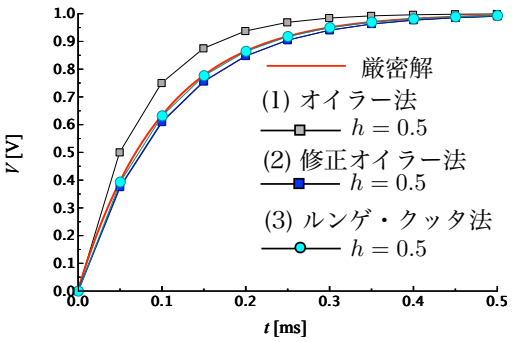
$$k_4 = f(x_{n+1}, y_n + hk_3)$$

$$y_{n+1} = y_n + \frac{h}{6}(k_1 + 2k_2 + 2k_3 + k4)$$

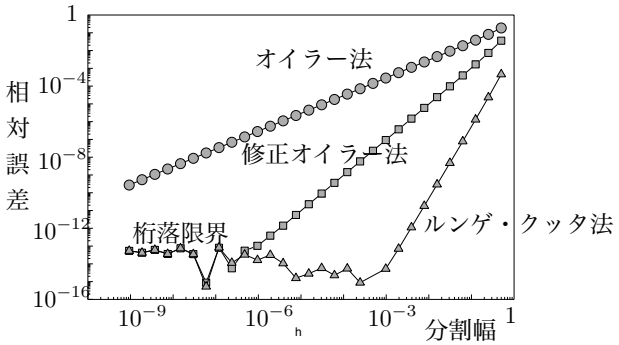
ルンゲ・クッタ法のプログラム

```
int main(void) {
  int i, n = 100;/* 繰り返し回数 */
  double x = 0.0, y = 0.0; /* 初期値 */
  double dx = 0.01;   /* 刻み幅 */
  double dy1, dy2, dy3, dy4;
  output(x, y);
  for (i=0; i<n; i++) {
    dy1 = dydx(x, y)*dx;
    dy2 = dydx(x + dx/2., y + dy1/2.)*dx;
    dy3 = dydx(x + dx/2., y + dy2/2.)*dx;
    dy4 = dydx(x + dx, y + dy3)*dx;
    x = x + dx;
    y = y + (dy1 + 2.0*(dy2 + dy3) + dy4)/6.0;
    output(x, y);
  }
  return 0;
}
```

例題1(ルンゲ・クッタ法)



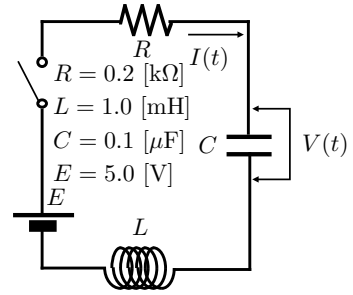
分割幅と誤差



$x = 1$のときの数値解の誤差

例題2

電圧$V(t)$と電流$I(t)$を計算する。



$$LC\frac{d^2}{dt^2}V(t) + RC\frac{d}{dt}V(t) + V(t) = E$$

$$I(t) = C\frac{d}{dt}V(t)$$

$$\tau_1 = \sqrt{LC} \quad \tau_2 = RC$$

$$x \equiv \frac{t}{\tau_1} \qquad y \equiv \frac{V}{E}$$

$$\frac{d^2y}{dx^2} + \frac{\tau_2}{\tau_1}\frac{dy}{dx} + y = 1$$

$$V(\tau_1 x) = Ey(x) \qquad I(\tau_1 x) = I_0\frac{dy(x)}{dx} \qquad \left(I_0 = E\sqrt{\frac{C}{L}}\right)$$

例題2

電圧$V(t)$と電流$I(t)$を計算する。

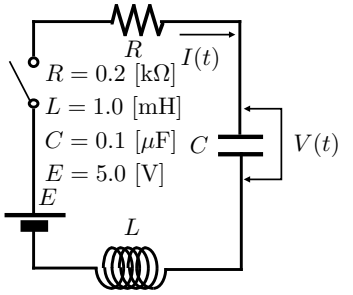
$$\tau_1 = \sqrt{LC}$$
$$= 1.0 \times 10^{-5}\ [\mathrm{s}] = 10\ [\mu\mathrm{s}]$$

$$\tau_2 = RC$$
$$= 2 \times 10^{-5}\ [\mathrm{s}] = 20\ [\mu\mathrm{s}]$$

$$E\sqrt{\frac{C}{L}} = 5.0 \times \sqrt{\frac{0.1 \times 10^{-6}}{1.0 \times 10^{-3}}}$$
$$= 50\ [\mathrm{mA}] \equiv I_0$$

例題2

電圧$V(t)$と電流$I(t)$を計算する。



$\dfrac{d^2y}{dx^2} + \dfrac{\tau_2}{\tau_1}\dfrac{dy}{dx} + y = 1$ を、

初期条件 $y(0) = 0$

$\dfrac{dy}{dx}(0) = 0$

の下で解き

$$V(\tau_1 x) = Ey(x)$$

$$I(\tau_1 x) = I_0\frac{dy(x)}{dx}$$

を出力する

例題2

$$\frac{d^2y}{dx^2} + \frac{\tau_2}{\tau_1}\frac{dy}{dx} + y = 1$$

$$\frac{d^2y}{dx^2} + 2\frac{dy}{dx} + y = 1 \qquad y(0) = 0 \quad \frac{dy}{dx}(0) = 0$$

$$\frac{dy}{dx} = z \qquad y(0) = 0$$

$$\frac{dz}{dx} = 1 - 2z - y \qquad z(0) = 0$$

$\tau_1 = 0.01$ [ms] $\quad E = 5$ [V] $\quad I_0 = 50$ [mA]

$$V(\tau_1 x) = Ey(x)$$

$$I(\tau_1 x) = I_0\frac{dy(x)}{dx}$$

オイラー法のプログラム

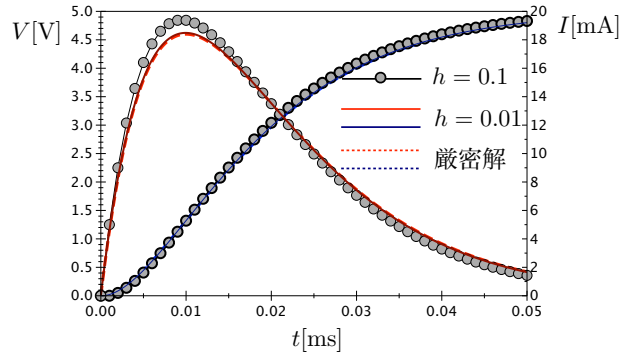
オイラー法のプログラム(2階)

```
#include <stdio.h>     /* 必要ならば math.h も */
double dydx(double x, double y, double z);
double dzdx(double x, double y, double z);
void output(double x, double y, double z);
int main(void) {
  int i, n = 100;      /* 繰り返し回数 */
  double x = 0.0, y = 0.0, z = 0.0; /* 初期値 */
  double dx = 0.01;    /* 刻み幅 */
  double dy, dz;
  output(x, y, z);
  for (i=0; i<n; i++) {
    dy = dydx(x, y, z)*dx;
    dz = dzdx(x, y, z)*dx;
    x = x + dx;
    y = y + dy;
    z = z + dz;
    output(x, y, z);
  }
  return 0;
}
```

オイラー法のプログラム(2)

```
double dydx(double x, double y, double z) {
  return z;
}
double dzdx(double x, double y, double z) {
  return (1.0 - 2.0*z - y);
}
void output(double x, double y, double z) {
  double E0 = 5.0;  /* V */
  double I0 = 50.0; /* mA */
  double T1 = 0.01; /* ms */
  printf ("%f %f %f\n", T1*x, E0*y, I0*z);
}
```

例題2(オイラー法・計算結果)

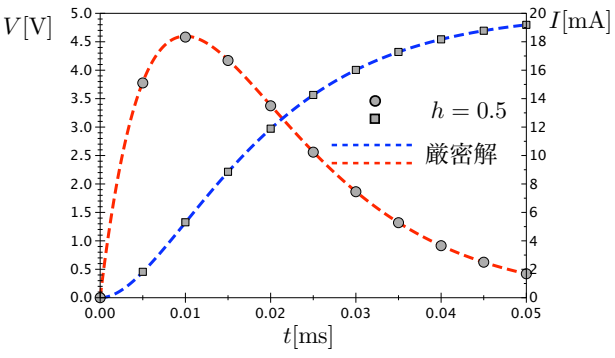


例題2(ルンゲ・クッタ法)

1階(1変数)　　2階(2変数連立)

```
dy1 = dydx(x, y, z)*dx;
dz1 = dzdx(x, y, z)*dx;
dy2 = dydx(x + dx/2., y + dy1/2., z +dz1/2.)*dx;
dz2 = dzdx(x + dx/2., y + dy1/2., z +dz1/2.)*dx;
dy3 = dydx(x + dx/2., y + dy2/2., z + dz2/2.)*dx;
dz3 = dzdx(x + dx/2., y + dy2/2., z + dz2/2.)*dx;
dy4 = dydx(x + dx, y + dy3, z + dz3)*dx;
dz4 = dzdx(x + dx, y + dy3, z + dz3)*dx;
x = x + dx;
y = y + (dy1 + 2.0*(dy2 + dy3) + dy4)/6.0;
z = z + (dz1 + 2.0*(dz2 + dz3) + dz4)/6.0;
```

例題2(ルンゲ・クッタ法)



特別付録

2階微分方程式共通

```
#include <stdio.h>
#include <math.h>
double dydx(double x, double y, double z);
double dzdx(double x, double y, double z);
int main(void) {
  int i, n = 100;
  double x = 0.0, y = 0.0, z = 0.0;
  double dx = 0.01;
  各アルゴリズムのプログラム
  return 0;
}
```

問題にあわせて変更する

2階微分方程式共通

```
double dydx(double x, double y, double z) {
  return z;
}
double dzdx(double x, double y, double z) {
  double ret;
  ret = 1.0 - 2.0*z - y;
  return ret;
}
```

問題にあわせて変更する

オイラー法

```
double dy, dz;
printf("%f %f %f\n", x, y, z);
for (i=0; i<n; i++) {
  dy = dydx(x, y, z)*dx;
  dz = dzdx(x, y, z)*dx;
  x = x + dx;
  y = y + dy;
  z = z + dz;
  printf("%f %f %f\n", x, y, z);
}
```

修正オイラー法

```
double dy1, dy2, dz1, dz2;
printf("%f %f %f\n", x, y, z);
for (i=0; i<n; i++) {
  dy1 = dydx(x, y, z)*dx;
  dz1 = dzdx(x, y, z)*dx;
  dy2 = dydx(x + dx, y + dy1, z +dz1)*dx;
  dz2 = dzdx(x + dx, y + dy1, z +dz1)*dx;
  x = x + dx;
  y = y + (dy1 + dy2)/2.0;
  z = z + (dz1 + dz2)/2.0;
  printf("%f %f %f\n", x, y, z);
}
```

ルンゲ・クッタ法

```
double dy1, dy2, dy3, dy4, dz1, dz2, dz3, dz4;
printf("%f %f %f\n", x, y, z);
for (i=0; i<n; i++) {
  dy1 = dydx(x, y, z)*dx;
  dz1 = dzdx(x, y, z)*dx;
  dy2 = dydx(x + dx/2., y + dy1/2., z +dz1/2.)*dx;
  dz2 = dzdx(x + dx/2., y + dy1/2., z +dz1/2.)*dx;
  dy3 = dydx(x + dx/2., y + dy2/2., z + dz2/2.)*dx;
  dz3 = dzdx(x + dx/2., y + dy2/2., z + dz2/2.)*dx;
  dy4 = dydx(x + dx, y + dy3, z + dz3)*dx;
  dz4 = dzdx(x + dx, y + dy3, z + dz3)*dx;
  x = x + dx;
  y = y + (dy1 + 2.0*(dy2 + dy3) + dy4)/6.0;
  z = z + (dz1 + 2.0*(dz2 + dz3) + dz4)/6.0;
  printf("%f %f %f\n", x, y, z);
}
```