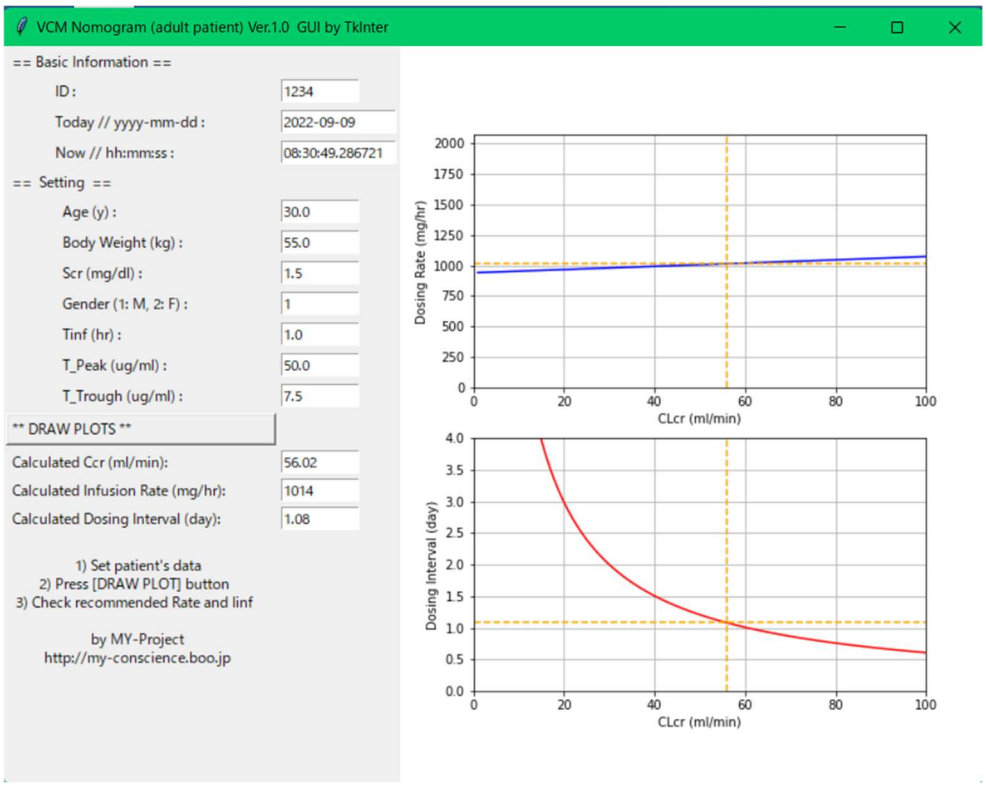
Python#009★★☆☆（VCM ノモグラム）　　2022.9.9

薬物動態～GUI 使って VCM ノモグラム

既に VCM のノモグラムについて自作してみたが、さらにデータ入力やグラフ表示をひとつの GUI 画面に組み込み、任意の条件での計算を行えるように工夫してみた。成人患者でのノモグラム画面を示す。（なお、この記事の結果を用いて生じるいかなる問題点についても筆者は責任を持たない）

Python コードを示した。



以上

```
#==
## VCM Nomogram for adult patients

import tkinter as tk
#import tkinter.ttk as ttk
import math
import datetime
import matplotlib.pyplot as plt
from matplotlib.backends.backend_tkagg import FigureCanvasTkAgg
#from matplotlib.figure import Figure

msg =    "¥n   1) Set patient's data¥n"
msg = msg + " 2) Press [DRAW PLOT] button¥n"
msg = msg + " 3) Check recommended Rate and Iinf"
msg = msg + "¥n¥n   by MY-Project¥n   http://my-conscience.boo.jp"

# Gloval Variables; defaults
Age1 = 30.0
Bwt1 = 55.0
```

```
Gender1 = 1 # Male
Scr1 = 1.5
Ccr = 0.0
Ccr1 = 0.0

Dint = 1.0
rate = 1000.
Dint2 = 1.0
rate2 = 1000.

Tinf = 1.0
Tpeak = 50.0
Ttrough = 7.5

x = []
y = []
R = []

fig = plt.Figure(figsize = (10, 10))

def plots():
    global Dint, rate, Dint2, rate2
    # Oragne Line
    Ccr1 = calc_ccr(Age1, Bwt1, Scr1, Gender1)
    Dint, rate = calc_adult(Ccr1)
    # Simulations
    for i in range(1, 101):
        x.append(i)
        Ccr = i
        Dint2, rate2 = calc_adult(Ccr)
        y.append(Dint2)
        R.append(rate2)
    return x, y, R

def draw_figures():
# Delete previus graphs
    ax1.cla()
    ax2.cla()

# - Dosing Rate
    #ax1 = fig.add_subplot(211)
    ax1.set_xlabel('CLcr (ml/min)')
    ax1.set_ylabel('Dosing Rate (mg/hr)')
    ax1.grid()
    R_max = max(R) + 1000 # Max of y-axis
    ax1.set_xlim(0, 100)
    ax1.set_ylim(0, R_max)
    ax1.plot(x, R, color = 'blue')
    ax1.vlines(Ccr1, 0, R_max, 'orange', 'dashed')
    ax1.hlines(rate, 0, 100, 'orange', 'dashed')

# - Dosing Interval
    #ax2 = fig.add_subplot(212)
    ax2.set_xlabel('CLcr (ml/min)')
    ax2.set_ylabel('Dosing Interval (day)')
    ax2.grid()
    #y_max = max(y) + 2
    ax2.set_xlim(0, 100)
    ax2.set_ylim(0, 4)
    ax2.plot(x, y, color = 'red')
    ax2.vlines(Ccr1, 0, Dint * 5, 'orange', 'dashed')
    ax2.hlines(Dint, 0, 100, 'orange', 'dashed')

    canvas.draw()

def calc_ccr(age, bwt, scr, gen):
    global Ccr1
    Ccr1 = (140 - age) * bwt / 72.0 / scr
```

```
    if gen == 2:
       Ccr1 = Ccr1 * 0.85
    return Ccr1

def calc_adult(Ccr):
    K12 = 0.525
    K21 = 0.213
    Vss = 60.7
    Vc = Vss / (1 + K12 / K21)
    CL = 0.0478 * Ccr
    Ke = CL / Vc
    temp1 = K12 + K21 + Ke
    temp2 = K21 * Ke
    alpha = 0.5 * (temp1 + math.sqrt(temp1 ** 2 - 4 * temp2))
    beta  = 0.5 * (temp1 - math.sqrt(temp1 ** 2 - 4 * temp2))
    aaa   = (alpha - K21) / alpha * (1 - math.exp(-alpha * Tinf))
    bbb   = (K21 -  beta) / beta  * (1 - math.exp(-beta  * Tinf))
    ccc   = aaa + Tpeak / Ttrough * bbb * math.exp(beta * Tinf)
    ccc   = ccc / (aaa + bbb)

    Dint2 = math.log(ccc) / beta / 24
    ddd   = Tpeak * math.exp(beta * Tinf) - Ttrough
    rate2 = ddd * Vc * (alpha - beta) / (aaa + bbb) / math.exp(beta * Tinf)
    return Dint2, rate2

def btn_go_clicked():
    global Age1, Bwt1, Gender1, Scr1, Ccr1
    global Tinf, Tpeak, Ttrough
    global x, y, R

    Age1 = float(ent_Age.get())
    Bwt1 = float(ent_BWT.get())
    Scr1 = float(ent_Scr.get())
    Gender1 = int(ent_Gen.get())
    Ccr1 = calc_ccr(Age1, Bwt1, Scr1, Gender1)
    ent_Ccr.delete(0, tk.END)  # Deleted
    ent_Ccr.insert(0, ('{:.2f}'.format(Ccr1)))

    Tinf = float(ent_Tinf.get())
    Tpeak = float(ent_TP.get())
    Ttrough = float(ent_TT.get())

    # reset plots' data
    x = []
    y = []
    R = []

    # calc. and draw plots' data
    plots()

    ent_R.delete(0, tk.END)  # Deleted
    ent_R.insert(0, ('{:.0f}'.format(rate)))
    ent_y.delete(0, tk.END)  # Deleted
    ent_y.insert(0, ('{:.2f}'.format(Dint)))

    ent_TODAY.delete(0, tk.END)  # Deleted
    ent_TODAY.insert(0, datetime.date.today())
    ent_NOW.delete(0, tk.END)  # Deleted
    ent_NOW.insert(0, datetime.datetime.now().time())

    draw_figures()

def _destroyWindow():
    root.quit()
    root.destroy()

## Main Routine ##
root = tk.Tk()
```

```
root.title("VCM Nomogram (adult patient) Ver.1.0  GUI by TkInter")
root.geometry("800x600")

#create_widgets()
#   - Create Main Windows (panedwin)
PW1 = tk.PanedWindow(root, orient = 'vertical')
PW1.pack(expand = True, fill = tk.BOTH, side = "left")

#  - Title Label
lbl_0 = tk.Label(PW1, anchor = "w", text = "== Basic Information ==",  width = 30)
lbl_0.grid(row = 0, column = 0, padx = 1, pady = 1)
#  - ID
lbl_ID = tk.Label(PW1, anchor = "w", text = "ID : ",  width = 20)
lbl_ID.grid(row = 1, column = 0, padx = 2, pady = 2)
ent_ID = tk.Entry(PW1, width = 10)
ent_ID.grid(row = 1, column = 1, sticky = tk.W,  padx = 2, pady = 2)
ent_ID.delete(0, tk.END)  # Deleted
#ent_ID.insert(0, "99999")
# - Today
lbl_TODAY = tk.Label(PW1, anchor = "w", text = "Today // yyyy-mm-dd : ",  width = 20)
lbl_TODAY.grid(row = 2, column = 0, padx = 2, pady = 2)
ent_TODAY = tk.Entry(PW1, width = 15)
ent_TODAY.grid(row = 2, column = 1, sticky = tk.W,  padx = 2, pady = 2)
ent_TODAY.delete(0, tk.END)  # Deleted
ent_TODAY.insert(0, datetime.date.today())
# - Now
lbl_NOW = tk.Label(PW1, anchor = "w", text = "Now // hh:mm:ss : ",  width = 20)
lbl_NOW.grid(row = 3, column = 0, padx = 2, pady = 2)
ent_NOW = tk.Entry(PW1, width = 15)
ent_NOW.grid(row = 3, column = 1, sticky = tk.W, padx = 2, pady = 2)
ent_NOW.delete(0, tk.END)  # Deleted
ent_NOW.insert(0, datetime.datetime.now().time())

#  - Setting
lbl_SET = tk.Label(PW1, anchor = "w", text = "==  Setting  ==",  width = 30)
lbl_SET.grid(row = 4, column = 0, padx = 1, pady = 1)
# - Age
lbl_Age = tk.Label(PW1, anchor = "w", text = "  Age (y) : ",  width = 20)
lbl_Age.grid(row = 5, column = 0, padx = 2, pady = 2)
ent_Age = tk.Entry(PW1, width = 10)
ent_Age.grid(row = 5, column = 1, sticky = tk.W,  padx = 2, pady = 2)
ent_Age.delete(0, tk.END)  # Deleted
ent_Age.insert(0, Age1)
# - Body Weight
lbl_BWT = tk.Label(PW1, anchor = "w", text = "  Body Weight (kg) : ",  width = 20)
lbl_BWT.grid(row = 6, column = 0, padx = 2, pady = 2)
ent_BWT = tk.Entry(PW1, width = 10)
ent_BWT.grid(row = 6, column = 1, sticky = tk.W,  padx = 2, pady = 2)
ent_BWT.delete(0, tk.END)  # Deleted
ent_BWT.insert(0, Bwt1)
# - Scr
lbl_Scr = tk.Label(PW1, anchor = "w", text = "  Scr (mg/dl) : ",  width = 20)
lbl_Scr.grid(row = 7, column = 0, padx = 2, pady = 2)
ent_Scr = tk.Entry(PW1, width = 10)
ent_Scr.grid(row = 7, column = 1, sticky = tk.W,  padx = 2, pady = 2)
ent_Scr.delete(0, tk.END)  # Deleted
ent_Scr.insert(0, Scr1)
# - Gender
lbl_Gen = tk.Label(PW1, anchor = "w", text = "  Gender (1: M, 2: F) : ",  width = 20)
lbl_Gen.grid(row = 8, column = 0, padx = 2, pady = 2)
ent_Gen = tk.Entry(PW1, width = 10)
ent_Gen.grid(row = 8, column = 1, sticky = tk.W,  padx = 2, pady = 2)
ent_Gen.delete(0, tk.END)  # Deleted
ent_Gen.insert(0, Gender1)
# - Infusion Time
lbl_Tinf = tk.Label(PW1, anchor = "w", text = "  Tinf (hr) : ",  width = 20)
lbl_Tinf.grid(row = 9, column = 0, padx = 2, pady = 2)
ent_Tinf = tk.Entry(PW1, width = 10)
```

```
ent_Tinf.grid(row = 9, column = 1, sticky = tk.W,  padx = 2, pady = 2)
ent_Tinf.delete(0, tk.END)  # Deleted
ent_Tinf.insert(0, Tinf)
# - Target Peak
lbl_TP = tk.Label(PW1, anchor = "w", text = "  T_Peak (ug/ml) : ",  width = 20)
lbl_TP.grid(row = 10, column = 0, padx = 2, pady = 2)
ent_TP = tk.Entry(PW1, width = 10)
ent_TP.grid(row = 10, column = 1, sticky = tk.W,  padx = 2, pady = 2)
ent_TP.delete(0, tk.END)  # Deleted
ent_TP.insert(0, Tpeak)
# - Target Trough
lbl_TT = tk.Label(PW1, anchor = "w", text = "  T_Trough (ug/ml) : ",  width = 20)
lbl_TT.grid(row = 11, column = 0, padx = 2, pady = 2)
ent_TT = tk.Entry(PW1, width = 10)
ent_TT.grid(row = 11, column = 1, sticky = tk.W,  padx = 2, pady = 2)
ent_TT.delete(0, tk.END)  # Deleted
ent_TT.insert(0, Ttrough)

# GO button
btn_go = tk.Button(PW1, anchor = "w", text = "** DRAW PLOTS **",
                command = btn_go_clicked, width = 30)
btn_go.grid(row = 13, column = 0, sticky = tk.W, padx = 2, pady = 2)
#  - Ccr
lbl_Ccr = tk.Label(PW1, anchor = "w", text = "Calculated Ccr (ml/min): ", width = 30)
lbl_Ccr.grid(row = 15, column = 0, padx = 1, pady = 1)
ent_Ccr = tk.Entry(PW1, width = 10)
ent_Ccr.grid(row = 15, column = 1, sticky = tk.W, padx = 2, pady = 2)
ent_Ccr.delete(0, tk.END)  # Deleted

#  - Calculated Rate
lbl_R = tk.Label(PW1, anchor = "w", text = "Calculated Infusion Rate (mg/hr): ", width =
30)
lbl_R.grid(row = 16, column = 0, padx = 1, pady = 1)
ent_R = tk.Entry(PW1, width = 10)
ent_R.grid(row = 16, column = 1, sticky = tk.W, padx = 2, pady = 2)
ent_R.delete(0, tk.END)  # Deleted

#  - Calculated Tinf
lbl_y = tk.Label(PW1, anchor = "w", text = "Calculated Dosing Interval (day): ", width =
30)
lbl_y.grid(row = 17, column = 0, padx = 1, pady = 1)
ent_y = tk.Entry(PW1, width = 10)
ent_y.grid(row = 17, column = 1, sticky = tk.W, padx = 2, pady = 2)
ent_y.delete(0, tk.END)  # Deleted

lbl_msg = tk.Label(PW1, anchor = "w", text = msg, width = 30)
lbl_msg.grid(row = 18, column = 0, padx = 1, pady = 1)

# Ploting Panels
Frame1 = tk.Frame(root)
Frame1.pack(side = tk.LEFT)
canvas = FigureCanvasTkAgg(fig, Frame1)
ax1 = fig.add_subplot(211)
ax2 = fig.add_subplot(212)
canvas.get_tk_widget().pack(side=tk.TOP, fill=tk.BOTH, expand=True)
plots()
draw_figures()

root.mainloop()
```