Australian Centre for Pharmacometrics

PKADVAN Package: Analytical Solutions in R

Australian Centre of Pharmacometrics
University of South Australia

Ahmad Abuhelwa
Email: Ahmad.Abuhelwa@mymail.unisa.edu.au

Objective

- Perform simulations in R using ADVAN-style analytical solutions
- Simulations examples
  - Example 1: 2-comp first-order absorption
  - Example 2: 2-comp intravenous infusion
  - Example 3: 2-comp intravenous bolus
  - Example 4: 2-comp 2-transit with time-varying covariate

ADVAN-style analytical solutions?

- Analytical solutions of linear compartmental models
- Used to simulate the time-course of drug amounts in each compartment of a pharmacokinetic systems.
- Derived using Laplace transform and coded in R/C++ languages
- They calculate the change in drug amounts in each compartment of the model over a time interval (t; $t = t_2 - t_1$) accounting for any dose or covariate events acting in the time interval.

Why ADVAN-style analytical solutions?

- Speed advantages
- Capacity to handle arbitrary dosing
- Capacity to handle time-varying covariates

PKADVAN - package

- Analytical solutions of 26 different PK models were derived, coded in R/C++, and incorporated into an R package “PKADVAN” package.
- All pharmacokinetic models solutions incorporated into the “PKADVAN” package have been validated against NONMEM
- GitHub: https://github.com/abuhelwa/PKADVAN_Rpackage

PKADVAN-package library

Category	Pharmacokinetic model	“PKADVAN” function name
Basic pharmacokinetic models	1 comp IV bolus	`OneCompIVbolus`
	2 comp IV bolus	`TwoCompIVbolus`
	3 comp IV bolus	`ThreeCompIVbolus`
	1 comp IV infusion	`OneCompIVinfusion`
	2 comp IV infusion	`TwoCompIVinfusion`
	3 comp IV infusion	`ThreeCompIVinfusion`
	1 comp first-order absorption	`OneCompFirstOrderAbs`
	2 comp first-order absorption	`TwoCompFirstOrderAbs`
	3 comp first-order absorption	`ThreeCompFirstOrderAbs`
Transit first-order absorption models	1 comp-1 transit	`OneCompOneTransit`
	1 comp-2 transit	`OneCompTwoTransit`
	1 comp-3 transit	`OneCompThreeTransit`
	1 comp-4 transit	`OneCompFourTransit`
	2 comp-1 transit	`TwoCompOneTransit`
	2 comp-2 transit	`TwoCompTwoTransit`
	2 comp-3 transit	`TwoCompThreeTransit`
	2 comp-4 transit	`TwoCompFourTransit`
First-order formation metabolite models	1 comp IV bolus-1 comp metabolite	`OneCompIVbolusOneCompMetab`
	2 comp IV bolus-1 comp metabolite	`TwoCompIVbolusOneCompMetab`
	3 comp IV bolus-1 comp metabolite	`ThreeCompIVbolusOneCompMetab`
	1 comp IV infusion-1 comp metabolite	`OneCompIVinfusionOneCompMetab`
	2 comp IV infusion-1 comp metabolite	`TwoCompIVinfusionOneCompMetab`
	3 comp IV infusion-1 comp metabolite	`ThreeCompIVinfusionOneCompMetab`
	1 comp first-order absorption-1 comp metabolite	`OneCompFirstOrderAbsOneCompMetab`
	2 comp first-order absorption-1 comp metabolite	`TwoCompFirstOrderAbsOneCompMetab`
	3 comp first-order absorption-1 comp metabolite	`ThreeCompFirstOrderAbsOneCompMetab`

Processing simulations

- Set up a NONMEM-style data frame with time sequence, dosing events & individual PK parameters → InputDataFrame
  - Columns for ID, TIME, AMT
  - Individual PK parameters of the respective model (e.g. CL, Q, V, etc.) including any covariate effects on the PK parameters.
- Use the PKADVAN function with 'ddply' to process simulation for each ID.
  - Returns original InputDataFrame with additional columns for:
    - Simulated drug amounts in each compartment
    - IPREDs in the central compartment
- Add residual variability on IPREDs

Example 1: first-order absorption model

- 2 compartment first-order absorption model
  - PPV on CL, central volume (V2) and KA
  - Proportional error model (with option for additive)
  - Gender, smoking & creatinine clearance effects on CL

Step 1: Define the PK model & Create a data frame with the individual PK parameters for *n* individuals.

- Define PK parameters which includes values for:
  - Population PK parameters (THETA's)
  - Covariates and between subject variability (ETA's) parameters

```
#-------------------------------------------------------------------------------
# Step 1: Setup a data frame with individual PK parameters including any covariate effects on PK parameters
#-------------------------------------------------------------------------------
    # Define individual
    n <- 1000  #Number of "individuals"
    ID <- seq(from = 1, to = n, by = 1)  #Simulation ID
    WT <- 70  #Total body weight, kg
    AGE <- 60  #Age, years
    SECR <- 100  #Serum Creatinine, umol/L
    SEX <- 0  #Gender, Male (0) Female (1)
    SMOK <- 0  #Smoking Status, Not Current (0) Current (1)

    # Define parameter values
    #Thetas
    CLPOP <- 10     #Clearance, L/h
    V2POP <- 50     #Volume of central compartment, L
    QPOP <-  10     #Inter-compartmental clearance, L/h
    V3POP <- 100    #Volume of peripheral compartment, L
    KAPOP <- 0.5    #Absorption rate constant, h^-1
    F1POP <- 1      #Bioavailability

    COV1 <- 0.5   #Effect of smoking status
    COV2 <- 1.15  #Effect of creatinine clearance on clearance

    #Omegas (as SD)
    ETA1SD <- 0.16
    ETA2SD <- 0.16
    ETA3SD <- 0.16
```

Step 1 cont'd

- Use random number generator to simulate residuals from a normal distribution

```
  #Omegas (as SD)
  ETA1SD <- 0.16
  ETA2SD <- 0.16
  ETA3SD <- 0.16

  #Specify a correlation matrix for ETA's
  R12 <- 0.5  #Correlation coefficient for CL-V1
  R13 <- 0.7  #Correlation coefficient for CL-KA
  R23 <- 0.5  #Correlation coefficient for V1-KA
# Calculate ETA values for each subject
  cor.vec <- c(
    1, R12, R13,
    R12, 1, R23,
    R13, R23, 1)
  CORR <- matrix(cor.vec, 3, 3)

# Specify the between subject variability for CL, V1, V2
  SDVAL <- c(ETA1SD, ETA2SD, ETA3SD)

# Use this function to turn CORR and SDVAL into a covariance matrix
  OMEGA <- cor2cov(CORR, SDVAL)

# Now use multivariate rnorm to turn the covariance matrix into ETA values
  ETAmat <- mvrnorm(n = n, mu = c(0, 0, 0), OMEGA)
  ETA1 <- ETAmat[, 1]
  ETA2 <- ETAmat[, 2]
  ETA3 <- ETAmat[, 3]
```

The 'mvrnorm' function uses the OMEGA matrix to generate random correlated ETA values

Step 1, cont'd

- Define individual parameter values including any covariate effects on the PK parameters and collect into a data frame 'par.data'.

```
97    # Define covariate effects
98    SMOKCOV <- 1
99    if(SMOK == 1) SMOKCOV <- SMOKCOV + COV1
100   CRCL <- ((140 - AGE)*WT)/(SECR*0.815)  # Male creatinine clearance
101   if(SEX == 1) CRCL <- CRCL*0.85  #Female creatinine clearance
102
103   # Define individual parameter values
104       # Note that 2 comp oral is parameterized using parameters: CL, V2, Q, V3, KA, F1 parameters
105       # refer to function documentation for more information
106   CL <- CLPOP*exp(ETA1)*((WT/70)^0.75)*SMOKCOV*((CRCL/90)^COV2)
107   V2 <- V2POP*exp(ETA2)*(WT/70)
108   Q  <- QPOP*(WT/70)^0.75
109   V3 <- V3POP*(WT/70)
110   KA <- KAPOP*exp(ETA3)
111   F1 <- F1POP
112
113   # Collect the individual parameter values in a parameter dataframe
114   par.data <- data.frame(
115       ID, CL, V2, Q, V3, KA, F1,  #patient parameters
116       WT, AGE, SECR, SEX, SMOK)  #covariates
117   head(par.data)
```

par.data

```
> # Collect the individual parameter values in a parameter dataframe
>       par.data <- data.frame(
+           ID, CL, V2, Q, V3, KA, F1,  #patient parameters
+           WT, AGE, SECR, SEX, SMOK)  #covariates
>       head(par.data, 10)
   ID        CL       V2  Q  V3        KA F1 WT AGE SECR SEX SMOK
1   1  8.598611 48.55601 10 100 0.5395161  1 70  60  100   0    0
2   2  6.666591 49.27283 10 100 0.4427415  1 70  60  100   0    0
3   3 10.098613 55.35602 10 100 0.5802440  1 70  60  100   0    0
4   4  6.801771 46.75425 10 100 0.4064813  1 70  60  100   0    0
5   5  7.846087 55.05923 10 100 0.5773364  1 70  60  100   0    0
6   6  4.863529 46.25984 10 100 0.4157334  1 70  60  100   0    0
7   7  6.655953 43.10783 10 100 0.4102701  1 70  60  100   0    0
8   8  7.000716 44.98235 10 100 0.4655221  1 70  60  100   0    0
9   9  6.787724 45.63072 10 100 0.4759928  1 70  60  100   0    0
10 10  6.838346 41.33173 10 100 0.4630459  1 70  60  100   0    0
```

Step 2: Create a NONME-style data frame with time sequence, dosing events and par.data

```
127     #Set dose records:
128     dosetimes <- c(seq(from = 0, to = 72, by = 12)) #Can be arbitrary [e,g: dosetimes <- c(0,6.5,12,48) ]
129
130     #Make a time sequence (hours). This should include dosetimes.
131     TIME <- sort(unique(c(seq(from = 0, to = 144, by = 0.25),dosetimes)))
132
133     #generate df with the time sequence
134     df <- expand.grid("ID"=ID,"TIME"=TIME,"AMT"=0,"MDV"=0)
135
136     #Set up doses in AMT column
137     doserows <- subset(df, TIME%in%dosetimes)   #subset dose rows (records at dosetimes)
138     doserows$AMT <- 500                         #Doses can be arbitrary
139     #doserows$AMT[doserows$TIME <= 24] <- 750
140     #doserows$AMT[doserows$TIME > 24]  <- 500
141     doserows$MDV <- 1
142
143     #Add back dose information
144     df <- rbind(df,doserows)
145
146 # Join par.data with the NONMEM-style data frame
147     inputDataFrame <- join(df, par.data,by="ID")
148
149 #This is crucial!
150     # Arrange "inputDataFrame" df by ID, TIME (ascending) and by AMT (descending)
151     inputDataFrame <- inputDataFrame[order(inputDataFrame$ID,inputDataFrame$TIME,inputDataFrame$AMT),]
152     # Remove extra row that has a TIME=0 and AMT=0
153     inputDataFrame <- subset(inputDataFrame, (TIME==0 & AMT==0)==F)
154     head(inputDataFrame, 10)
```

Creates a data frame from all combinations of the supplied vectors

rbind: combines two data frames objects by rows

'join' function from 'plyr' package: joins two data frames by a common variable name; ID

InputDataFrame

```
head(inputDataFrame, 10)
ID TIME AMT MDV       CL       V2  Q  V3       KA F1 WT AGE SECR SEX SMOK
 1 0.00 500   1 6.159954 42.15801 10 100 0.412882  1 70  60  100   0    0
 1 0.25   0   0 6.159954 42.15801 10 100 0.412882  1 70  60  100   0    0
 1 0.50   0   0 6.159954 42.15801 10 100 0.412882  1 70  60  100   0    0
 1 0.75   0   0 6.159954 42.15801 10 100 0.412882  1 70  60  100   0    0
 1 1.00   0   0 6.159954 42.15801 10 100 0.412882  1 70  60  100   0    0
 1 1.25   0   0 6.159954 42.15801 10 100 0.412882  1 70  60  100   0    0
 1 1.50   0   0 6.159954 42.15801 10 100 0.412882  1 70  60  100   0    0
 1 1.75   0   0 6.159954 42.15801 10 100 0.412882  1 70  60  100   0    0
 1 2.00   0   0 6.159954 42.15801 10 100 0.412882  1 70  60  100   0    0
 1 2.25   0   0 6.159954 42.15801 10 100 0.412882  1 70  60  100   0    0
```

Step 3 & 4 : Apply 'PKADVAN' function & Add residual variability on IPRED

```
155 #-------------------------------------------------------------
156 # step 3: Apply PKADVAN function of the respective PK model
157 #-------------------------------------------------------------
158     #PKADVAN functions retruns the inputDataFrame with calculated amounts in each compartment and IPREDs in the central compartment
159     sim.data <- ddply(inputDataFrame, .(ID), TwoCompFirstOrderAbs)
160     head(sim.data, 10)
161
162 #-------------------------------------------------------------
163 # step 4: Add residual unexplained variability to IPREDs
164 #-------------------------------------------------------------
165 # Use random number generator to simulate residuals from a normal distribution
166     #no. of observations =  no. of time points
167     nobs <- length(sim.data$TIME)
168     EPS1 <- rnorm(nobs, mean = 0, sd = EPS1SD)  #Proportional residual error
169     EPS2 <- rnorm(nobs, mean = 0, sd = EPS2SD)  #Additive residual error
170     sim.data$DV <- sim.data$IPRED*(1 + EPS1) + EPS2
171     head(sim.data,10)
```

'ddply' function applies 'TwoCompFirstOrderAbs' function for each ID

Calculate simulated DV values

Simulated data

```
>       head(sim.data,10)
   ID TIME AMT MDV       CL       V2  Q  V3       KA F1 WT AGE SECR SEX SMOK       A1        A2       A3    IPRED       DV
1   1 0.00 500   1 6.159954 42.15801 10 100 0.412882  1 70  60  100   0    0 500.0000   0.00000 0.000000 0.000000 0.000000
2   1 0.25   0   0 6.159954 42.15801 10 100 0.412882  1 70  60  100   0    0 450.9640  46.73295 1.420538 1.108519 1.329409
3   1 0.50   0   0 6.159954 42.15801 10 100 0.412882  1 70  60  100   0    0 406.7371  84.67788 5.277709 2.008583 2.141606
4   1 0.75   0   0 6.159954 42.15801 10 100 0.412882  1 70  60  100   0    0 366.8476 115.13864 11.035927 2.731121 1.725288
5   1 1.00   0   0 6.159954 42.15801 10 100 0.412882  1 70  60  100   0    0 330.8702 139.24414 18.243933 3.302910 4.714047
6   1 1.25   0   0 6.159954 42.15801 10 100 0.412882  1 70  60  100   0    0 298.4211 157.97052 26.523132 3.747105 3.945440
7   1 1.50   0   0 6.159954 42.15801 10 100 0.412882  1 70  60  100   0    0 269.1544 172.16064 35.557435 4.083699 4.541062
8   1 1.75   0   0 6.159954 42.15801 10 100 0.412882  1 70  60  100   0    0 242.7579 182.54110 45.084424 4.329927 4.011906
9   1 2.00   0   0 6.159954 42.15801 10 100 0.412882  1 70  60  100   0    0 218.9501 189.73729 54.887670 4.500622 6.471643
10  1 2.25   0   0 6.159954 42.15801 10 100 0.412882  1 70  60  100   0    0 197.4773 194.28645 64.790061 4.608529 7.163840
```

Examples 2 & 3: IV bolus/ infusion

- Using the same PK Model: 2-compartment model
- Run examples 2 and 3 from the R scripts for IV bolus/infusion
- Note the following:
  - 2 compartment IV models are parametrized using: CL, V1, Q, V2
  - For IV infusion: Infusion rate (RATE) must be added to the NONMEM-style data frame in addition to AMT.

Example 4: Time-varying covariates

- 2 compartment 2-transit absorption model
- Creatinine clearance (CLCR) as time-changing covariate on central clearance.
  - Simulate 100 mg oral dose
  - CLCR was deliberately changed from 100 ml/min (Time < 24h) to 30 ml/min (TIME >= 24 h)

Define PK model parameters

```
#----------------------
# Define PK parameters
#----------------------
#Define between subject variability on PK parameters
    #BSV (Omegas as SD)
    ETA1CL  <- 0 #0.15
    ETA2V2  <- 0 #0.12
    ETA3Q   <- 0 #0.14
    ETA4V3  <- 0 #0.05
    ETA5KTR <- 0 #0.30

#Define residual error model (Epsilons as SD)
    EPS1    <- 0        #Proportional residual error
    EPS2    <- 0        #Additive residual error

#Use random number generator to simulate residuals from a normal distribution
    BSVCL   <- rnorm(n, mean = 0, sd = ETA1CL)  #BSV on CL
    BSVV2   <- rnorm(n, mean = 0, sd = ETA2V2)  #BSV on V2
    BSVQ    <- rnorm(n, mean = 0, sd = ETA3Q)   #BSV on Q
    BSVV3   <- rnorm(n, mean = 0, sd = ETA4V3)  #BSV on V3
    BSVKTR  <- rnorm(n, mean = 0, sd = ETA5KTR) #BSV on KTR

#Set population PK parameters for 2-compartment 2-transit absorption model
    CLpop   <- 0.5      #clearance
    V2pop   <- 20       #central volume of distribution
    Qpop    <- 1        #inter-compartmental clearance
    V3pop   <- 25       #peripheral volume of distribution
    KTRpop  <- 2.05     #first-order absorption rate constant
    F1pop   <- 0.80     #Bioavailability
```

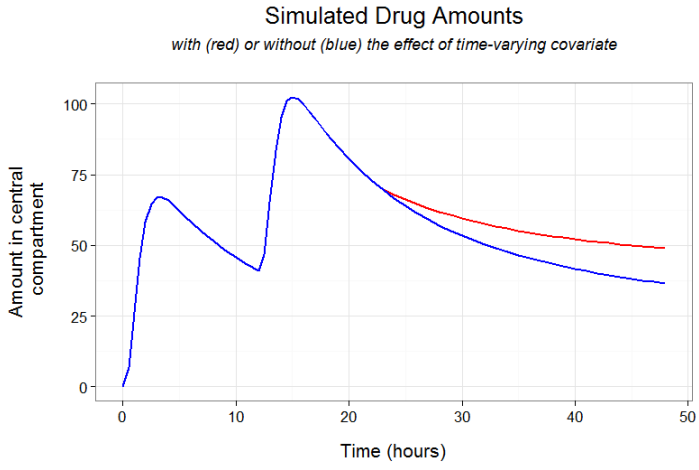
Generate a NONMEM-style data frame

```
#-----------------------------------------------------------
# Create a NONMEM-style data frame with dosing records
#-----------------------------------------------------------
    #Set number of subjects
    n <- 1
    ID <- 1:n

    #Set dose records:
    dosetimes <- c(0,12)  # This can be arbitrary

    #Now define finer sample times for after a dose to capture Cmax
    doseseq <- c(0,0.5,1,1.5,2,2.5,3,3.5,4,4.5,5,5.5,6,7,8,9,10)

    #Use the outer product but with addition to expand this doseseq for all dosetimes
    PKtimes <- outer(dosetimes,doseseq, FUN="+")

    #Make dataframe
    df <- expand.grid("ID"=ID,"TIME"=sort(unique(c(seq(0,48,1),PKtimes))),"AMT"=0,"MDV"=0,"CLCR"=NA)

    #Set time-varying creatinine clearance
    df$CLCR[df$ID ==1 & df$TIME <  24 ] <- 100
    df$CLCR[df$ID ==1 & df$TIME >= 24 ] <- 30

    #Set Doserows. It can be any arbitrary dose
    doserows <- subset(df, TIME%in%dosetimes)
    doserows$AMT <- 100
    doserows$MDV <- 1

    #Add back dose information
    df <- rbind(df,doserows)
```

Calculate individual PK parameters

```
#Modify df for PKADVAN calculations and include any covariates on PK parameters
    inputDataFrame <- df

#Calculate group parameter values including any covariate effects
    inputDataFrame$CL   <- CLpop*exp(BSVCL)*(inputDataFrame$CLCR/100)  #creatinine clearance (CLCR) added as a time-changing covariate on CL
    inputDataFrame$V2   <- V2pop*exp(BSVV2)
    inputDataFrame$Q    <- Qpop*exp(BSVQ)
    inputDataFrame$V3   <- V3pop*exp(BSVV3)
    inputDataFrame$KTR  <- KTRpop*exp(BSVKTR)
    inputDataFrame$F1   <- F1pop

#This is crucial!
    # Arrange "inputDataFrame" df by ID, TIME (ascending) and by AMT (descending)
    inputDataFrame <- inputDataFrame[order(inputDataFrame$ID,inputDataFrame$TIME,inputDataFrame$AMT),]
    # Remove extra row that has a TIME=0 and AMT=0
    inputDataFrame <- subset(inputDataFrame, (TIME==0 & AMT==0)==F)
    head(inputDataFrame, 10)

#Apply PKADVAN functions
    sim.data <- ddply(inputDataFrame, .(ID), TwoCompTwoTransit)  #simulated data with CLCR as time-varying covariate
    head(sim.data)

#simulated data Without accounting for the time-varying covariate
    inputDataFrame$CL   <- CLpop*exp(BSVCL)
    sim.data2 <- ddply(inputDataFrame, .(ID), TwoCompTwoTransit) #simulated data without CLCR covariate effect
    head(sim.data2)
```

Generate plot



Limitations

- Solving for analytical solutions using Laplace transforms will increase in complexity as the number of states in the pharmacokinetic system increases.

To do list

- Implement steady-state functionality, as achieved with the SS and II data items in NONMEM.
- Implement analytical solutions for combined dosing regimens (e.g., IV bolus plus infusion).

More examples & info

- https://github.com/abuhelwa/PKADVAN_Rpackage