链滴

descem

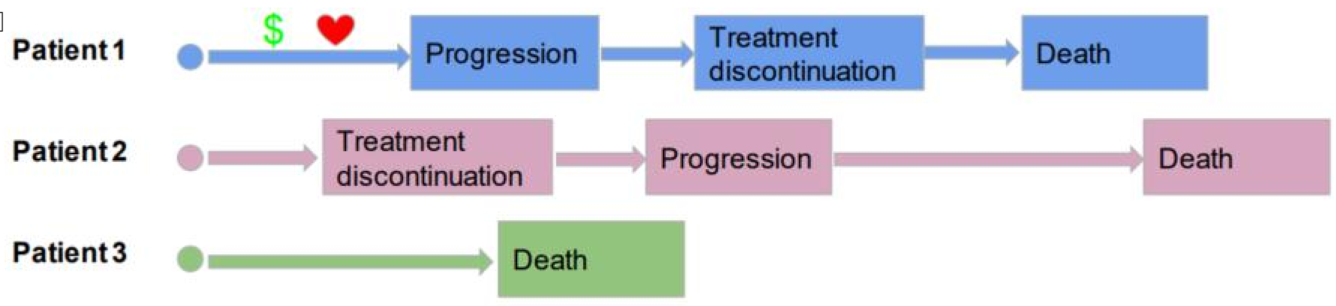
https://roche.github.io/descem/articles/example_eBC.html

原理

模型系统基于一系列事件流（如疾病恶化、进展、终止治疗等）的发生时间

事件以离散的间隔在时间轴上向前移动

患者以相关的属性信息（如接受不同的治疗）作为个体进入模拟



核心步骤

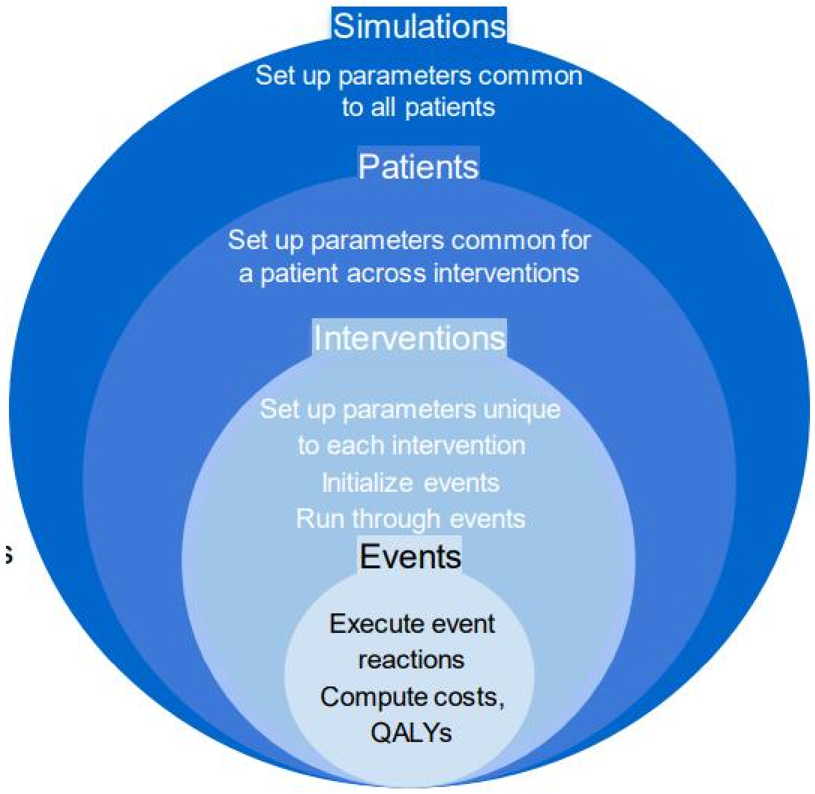
- 参数确定（Parameters）
  - 模拟期间的共同参数（common within simulation，如费用cost）
  - 不同治疗间患者的共同参数（common for a patient across interventions，如患者特征）
  - 每个患者和治疗的特定参数（specific to each patient and intervention）
- 初始事件和到事件的时间（Initial events and time to event）
- 事件关系（相互作用）的指定（Declaration of reaction to each event）
- 效用和费用（Utilities and costs(optional)）
- 运行模型和结果输出

模型引擎：循环描述和执行顺序

四重循环

- 每次模拟（PSA）
- 每个患者
- 每种治疗
- 每个事件

对于每个事件，计算先前和当前事件之间的折扣qalys/lys/成本，然后添加/更新反应中定义的事件/目


Simulations
Set up parameters common to all patients
Patients
Set up parameters common for a patient across interventions
Interventions
Set up parameters unique to each intervention
Initialize events
Run through events
Events
Execute event reactions
Compute costs, QALYs

参数说明

一般输入

```
#Put objects here that do not change on any patient or intervention loop
common_all_inputs <-add_item(
                      util.sick = 0.8,
                      util.sicker = 0.5,
                      cost.sick = 3000,
                      cost.sicker = 7000,
                      cost.int = 1000,
                      coef_noint = log(0.2),
                      HR_int = 0.8)

# Utilities:效用，健康=1.0，sick=0.8，sicker=0.5，death=0

#Put objects here that do not change as we loop through treatments for a patient
common_pt_inputs <- add_item(death= max(0.0000001,rnorm(n=1, mean=12, sd=3)))

#Put objects here that change as we loop through treatments for each patient (e.g. events can
affect fl.tx, but events do not affect nat.os.s)
unique_pt_inputs <- add_item(fl.sick = 1)
```

add_item()

定义模型中用于计算的参数。

`add_item(.data = NULL, ...)`

```
add_item(util_idfs = if(psa_bool){rnorm(1,0.8,0.2)} else{0.8},
         util.mbc = 0.6,
         cost_idfs = 2500)
```

add_tte()

定义事件和初始事件时间，每种治疗分开定义。

`add_tte(.data = NULL, trt, evts, other_inp = NULL, input)`

`trt`：事件对应的干预/治疗

`evts`：事件向量，需要通过`add_reactevt()`向事件添加变化及需要的计算

`other_inp`：模拟期间要保存的其他变量

`input`：evts参数中列出事件初始事件时间

```
init_event_list <-
  add_tte(trt="noint", evts = c("sick","sicker","death") ,input={
    sick <- 0
    sicker <- draw_tte(1,dist="exp", coef1=coef_noint)}) %>%
  add_tte(trt="int", evts = c("sick","sicker","death") ,input={
    sick <- 0
    sicker <- draw_tte(1,dist="exp", coef1=coef_noint, hr = HR_int)})
```

在init_event_list中定义2种治疗，每种治疗3个事件，`c("sick","sicker","death")`

`trt="noint"`中`sick`的初始时间为0，`sicker`的初始时间服从参数为coef_noint（`log(0.2)`）指数分布

`trt="int"`中`sick`的初始时间为0，`sicker`的初始时间服从HR=HR_int（0.8）的相对于参数为coef_noint（`log(0.2)`）指数分布

draw_tte()

通过一系列生存函数参数刻画事件时间。

`draw_tte(n_chosen = 1,dist = "exp",coef1 = 1,coef2 = NULL,coef3 = NULL,hr = 1,seed = NULL)` n_chosen：observations数

dist：分布类型，可以有'lnorm','weibullPH','weibull','llogis','gompertz','gengamma','gamma','e
p'

coef1：分布的第一个系数，在coef()中定义，在flexsurvreg对象中输出

coef2：分布的第二个系数

coef3：分布的第三个系数

hr：风险比

seed：模拟用种子数

draw_tte(n_chosen=1,dist='exp',coef1=1,hr=1)

add_reactevt()

定义事件发生对其他事件、费用、效用或item的影响。

add_reactevt(.data = NULL, name_evt, input)

name_evt：发生reactions的事件名称

input：事件发生时reactions的表达式

- modify_item()：增加、修改 items/flags/variables
- new_event()：添加事件
- modify_event()：用过时间修改已存在的事件

另外有标准变量

curtime：当前时间

prevtime：前一个事件的时间

cur_evtlist：患者尚未发生的事件

evt：当前执行的事件

i：患者号

simulation：正在迭代的模拟

模型将一直运行直到curtime设为Inf，因此终止模型的事件应该修改curtime并将其设置为Inf。

建议将添加/修改的一类inputs/events作为一个list以group形式在函数中处理，而不是一个一个单独
理。

添加事件（new_event）/修改事件（modify_event）时，必须定义或指明事件的名称和发生时间。

```
evt_react_list <-
  add_reactevt(name_evt = "sick",
               input = {}) %>%
  add_reactevt(name_evt = "sicker",
               input = {modify_item(list(fl.sick = 0))}) %>%
  add_reactevt(name_evt = "death",
               input = {modify_item(list(curtime = Inf))})
```

add_util()

定义事件和治疗的效用。

`add_util(.data = NULL, util, evt, trt, cycle_l = NULL, cycle_starttime = 0)`

`util`：用来计算效用估计的值或表达式

`evt`：效用对应的事件

`trt`：效用对应的治疗

`cycle_l`：Cycle长度，效用按cycle计算时会用到

`cycle_starttime`：效用开始计算的时间，效用按cycle计算时会用到

```
util_ongoing <- add_util(evt = c("sick", "sicker","death"),
                         trt = c("int", "noint"),
                         util = util.sick * fl.sick + util.sicker * (1-fl.sick))
```

add_cost()

定义事件和治疗的费用。

`add_cost(.data = NULL, cost, evt, trt, cycle_l = NULL, cycle_starttime = 0)`

`cost`：用来计算费用估计的值或表达式

`evt`：费用对应的事件

`trt`：费用对应的治疗

`cycle_l`：Cycle长度，费用按cycle计算时会用到

`cycle_starttime`：费用开始计算的时间，费用按cycle计算时会用到

RunSim()

运行模拟。

```
RunSim(
  trt_list = c("int", "noint"),
  common_all_inputs = NULL,
  common_pt_inputs = NULL,
  unique_pt_inputs = NULL,
  init_event_list = NULL,
  evt_react_list = evt_react_list,
  util_ongoing_list = NULL,
  util_instant_list = NULL,
  util_cycle_list = NULL,
  cost_ongoing_list = NULL,
  cost_instant_list = NULL,
  cost_cycle_list = NULL,
  npats = 500,
  n_sim = 1,
  psa_bool = NULL,
  ncores = 1,
  drc = 0.035,
  drq = 0.035,
  input_out = NULL,
  ipd = TRUE,
  debug = FALSE
)
```

trt_list：治疗的向量

common_all_inputs：患者总体参数

common_pt_inputs：治疗间患者参数（not affected by the intervention）

unique_pt_inputs：治疗内患者参数（change across each intervention）

init_event_list：初始事件和时间

evt_react_list：事件交互列表

util_ongoing_list：持续事件的效用列表（at an ongoing basis）

util_instant_list：临时事件的效用列表（accrued instantaneously at an event）

util_cycle_list：cycles内效用

cost_ongoing_list：持续事件的费用列表（at an ongoing basis）

cost_instant_list：临时事件的费用列表（accrued instantaneously at an event）

cost_cycle_list：cycles内费用

npats：模拟的患者数

n_sim：每个患者模拟数

psa_bool：是否执行PSA

drc：成本折扣率

drq：LYs/QALYs的折扣率

ipd：用于确定是否应返回单个患者数据的布尔值，如果设置为false，则只返回主要的聚合输出（微加快代码速度）

debug：用于确定是否应使用非并行RunEngine函数，这有助于调试

```
results <- RunSim(
  npats=1000,                              # number of patients to be simulated
  n_sim=1,                                 # number of simulations to run
  psa_bool = FALSE,                        # use PSA or not. If n_sim > 1 and psa_bool = FALSE, then
ifference in outcomes is due to sampling (number of pats simulated)
  trt_list = c("int", "noint"),            # intervention list
  common_all_inputs = common_all_inputs,   # inputs common that do not change within a s
mulation
  common_pt_inputs = common_pt_inputs,     # inputs that change within a simulation but ar
 not affected by the intervention
  unique_pt_inputs = unique_pt_inputs,     # inputs that change within a simulation between i
terventions
  init_event_list = init_event_list,       # initial event list
  evt_react_list = evt_react_list,         # reaction of events
  util_ongoing_list = util_ongoing,
  cost_ongoing_list = cost_ongoing,
  ncores = 2,                              # number of cores to use, recommended not to use all
  drc = 0.035,                             # discount rate for costs
  drq = 0.035                              # discount rate for qaly/lys
)
```

```
####################################################################
           int    noint
costs 54560.04 52079.85
lys       9.69     9.69
qalys     6.17     6.03
ICER        NA      Inf
ICUR        NA 17043.56
```

summary_results_det()

summary_results_psa()

summary_results_det(out = final_output, trt = NULL)

summary_results_psa(out = output_psa, trt = NULL)会得到置信区间

out：RunSim()的结果中的final_output数据

trt：设定参照治疗

```
summary_results_det(results$final_output)
summary_results_psa(results$output_psa)
```

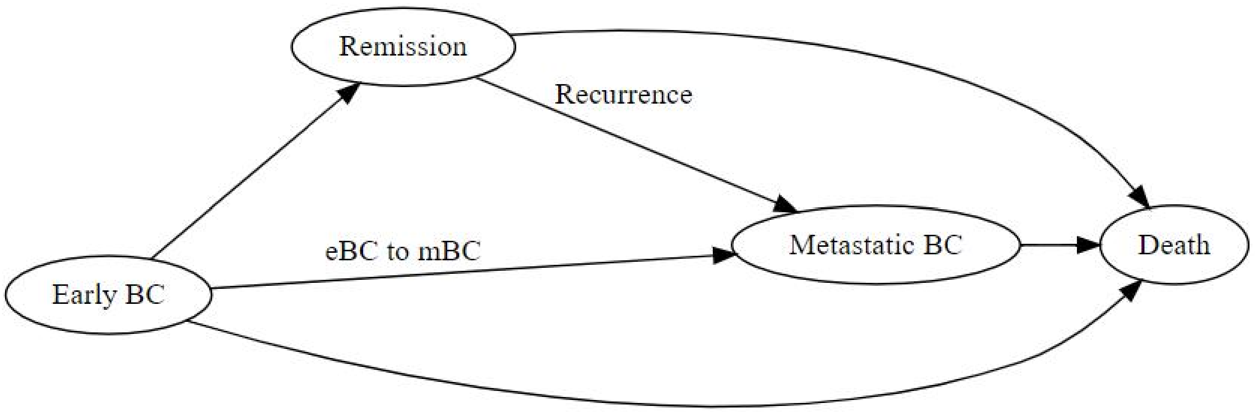
原文链接：descem

```
psa_ipd <- bind_rows(map(results$output_psa, "merged_df"))
psa_ipd[1:10,] %>%
  kable() %>%
  kable_styling(bootstrap_options = c("striped", "hover", "condensed", "responsive"))
```

```
 evtname          evttime         cost        qaly          ly          pat_id        trt
otal_costs          total_qalys         total_lys        simulation
 sick        0.0000000          0.000        0.0000000         0.0000000           1
nt          73626.16         5.475675         10.698576          1
 sicker         0.4243738         1685.164         0.3370329         0.4212911          1
nt          73626.16         5.475675         10.698576          1
 death         13.3406999         71940.994         5.1386424          10.2772848          1
nt          73626.16         5.475675         10.698576          1
 sick        0.0000000          0.000        0.0000000         0.0000000           2
nt          33429.94         4.655493         6.665406        1
 sicker         4.7819160         17637.206         3.5274411          4.4093014          2
nt          33429.94         4.655493         6.665406        1
 death         7.5710636         15792.730         1.1280521          2.2561043          2
nt          33429.94         4.655493         6.665406        1
 sick        0.0000000          0.000        0.0000000         0.0000000           3
nt          77087.01         6.584838         11.911283          1
 sicker         2.1768321         8389.288         1.6778575         2.0973219          3
nt          77087.01         6.584838         11.911283          1
 death         15.3259546         68697.724         4.9069803          9.8139605          3
nt          77087.01         6.584838         11.911283          1
 sick        0.0000000          0.000        0.0000000         0.0000000           4
nt          41007.00         8.201399         10.251749          1
```

示例

Model Concept



一般输入

```
#Utilities
util.data <- data.frame(name = c("util.idfs.ontx" ,"util.idfs.offtx" ,"util.remission" ,"util.recurren
e" ,"util.mbc.progression.mbc" ,"util.mbc.pps"),
                        value = c(0.75, 0.8,0.9,0.7,0.6,0.5),
                        se=rep(0.02,6),
                        stringsAsFactors = FALSE)

cost.data <- data.frame(name = c("cost.idfs.tx" ,"cost.recurrence" ,"cost.mbc.tx" ,"cost.tx.beva"
"cost.idfs.txnoint",
                                 "cost.idfs","cost.mbc.progression.mbc","cost.mbc.pps","cost.2ndline","cost
ae"),
                        value = c(40000,5000,3000,10000,30000,10000,20000,30000,20000,1000),string
AsFactors = FALSE) %>%
  mutate(se= value/5)
```

定义模型模拟需要的初始inputs和flag，可以定义

- 针对所有患者的输入（ common_all_inputs）
- 某种治疗中的特定独立患者（ common_pt_inputs）
- 特定治疗的特定患者（ unique_pt_inputs）

通过add_item函数添加item，并在之后的items中使用。所有输入都在执行事件和对事件的反应之定义完成。

程序首先执行common_all_inputs，之后common_pt_inputs，然后unique_pt_inputs。

fl.remissionflag刻画了概率为0.8的伯努利分布，表示80%的患者remission，20%的患者进展为ear
y metastatic BC。也可以使用至remission和至进展为mBC两种状态的的较短时间来建模。

定义状态（utilities）和cost的vector，之后分配给每个指定的item。

模型中运行的items（如util.remission）是unnamed。It is strongly recommended to assign unn
med objects if they are going to be processed in the model. In this case, we’ re only using uti
_v and cost_v as an intermediate input and these objects will not be processed。

```
#Each patient is identified through "i"
#Items used in the model should be unnamed numeric/vectors! otherwise if they are process
d by model it can lead to strangely named outcomes
#In this case, util_v is a named vector, but it's not processed by the model. We extract unnam
d numerics from it.

#Put objects here that do not change on any patient or intervention loop
common_all_inputs <- add_item( #utilities
  util_v = if(psa_bool){
    setNames(MASS::mvrnorm(1,util.data$value,diag(util.data$se^2)),util.data$name) #in this c
se I choose a multivariate normal with no correlation
  } else{setNames(util.data$value,util.data$name)},
  util.idfs.ontx = util_v[["util.idfs.ontx"]],
  util.idfs.offtx  = util_v[["util.idfs.offtx"]],
  util.remission = util_v[["util.remission"]],
  util.recurrence = util_v[["util.recurrence"]],
  util.mbc.progression.mbc = util_v[["util.mbc.progression.mbc"]],
  util.mbc.pps = util_v[["util.mbc.pps"]]
) %>%
  add_item( #costs
    cost_v = if(psa_bool){
      setNames(draw_gamma(cost.data$value,cost.data$se),cost.data$name) #in this case I cho
se a gamma distribution
    } else{setNames(cost.data$value,cost.data$name)},
    cost.idfs.tx  = cost_v[["cost.idfs.tx"]],
    cost.recurrence  = cost_v[["cost.recurrence"]],
    cost.mbc.tx  = cost_v[["cost.mbc.tx"]],
    cost.tx.beva  = cost_v[["cost.tx.beva"]],
    cost.idfs.txnoint = cost_v[["cost.idfs.txnoint"]],
    cost.idfs = cost_v[["cost.idfs"]],
    cost.mbc.progression.mbc = cost_v[["cost.mbc.progression.mbc"]],
    cost.mbc.pps = cost_v[["cost.mbc.pps"]],
    cost.2ndline = cost_v[["cost.2ndline"]],
    cost.ae = cost_v[["cost.ae"]]
  )

#Put objects here that do not change as we loop through interventions for a patient
common_pt_inputs <- add_item(sex_pt = ifelse(rbernoulli(1,p=0.01),"male","female"),
                             nat.os.s = draw_resgompertz(1,
                                                         shape=if(sex_pt=="male"){0.102}else{0.115},
                                                         rate=if(sex_pt=="male"){0.000016}else{0.0000041},
                                                         lower_bound = 50) ) #in years, for a patient who is 50yo

#Put objects here that change as we loop through treatments for each patient (e.g. events can
affect fl.tx, but events do not affect nat.os.s)
#common across trt but changes per pt could be implemented here (if (trt==)... )
unique_pt_inputs <- add_item(
  fl.idfs.ontx              = 1,
  fl.idfs                 = 1,
  fl.mbcs.ontx              = 1,
  fl.mbcs.progression.mbc   = 1,
  fl.tx.beva                = 1,
  fl.mbcs                   = 0,
  fl.mbcs_2ndline            = 0,
  fl.recurrence             = 0,
  fl.remission              = rbernoulli(1,0.8) #80% probability of going into remission
)
```

事件（Events）

添加初始事件

通过add_tte函数添加事件，每种治疗使用一次add_tte函数。

需要定义多个参数：

● one to indicate the **intervention**

● one to define the names of the **events** used

● one to define the names of **other objects** created that would like to store (optional, ma
be we generate an intermediate input which is not an event but that we want to save)

事件和其他对象将自动初始化为Inf。

init_event_list对象是通过使用add_tte函数两次来填充的，一个是针对“int”策略，另一个则是对“noint”策略。首先声明开始时间为0。

通过draw_tte()函数生成目标事件的发生时间

示例中事件的初始list为start, ttot, ttot.beva, progression.mbc, os, idfs, ttot.early, remission, recurrence and start.early.mbc。其他非初始化的事件可以在reactions part定义。

```
init_event_list <-
  add_tte(trt="int",
          evts = c("start","ttot", "ttot.beva","progression.mbc", "os","idfs","ttot.early","remission","
ecurrence","start.early.mbc"),
          other_inp = c("os.early","os.mbc"),
          input={ #intervention
            start <- 0

            #Early
            idfs <- draw_tte(1,'lnorm',coef1=2, coef2=log(0.2))
            ttot.early <- min(draw_tte(1,'lnorm',coef1=2, coef2=log(0.2)),idfs)
            ttot.beva <- draw_tte(1,'lnorm',coef1=2, coef2=log(0.2))
            os.early <- draw_tte(1,'lnorm',coef1=3, coef2=log(0.2))
            #if patient has remission, check when will recurrence happen
            if (fl.remission) {
              recurrence <- idfs +draw_tte(1,'lnorm',coef1=2, coef2=log(0.2))
              remission <- idfs

              #if recurrence happens before death
              if (min(os.early,nat.os.s)>recurrence) {
                #Late metastatic (after finishing idfs and recurrence)
                os.mbc <- draw_tte(1,'lnorm',coef1=0.8, coef2=log(0.2)) + idfs  +  recurrence
                progression.mbc <- draw_tte(1,'lnorm',coef1=0.5, coef2=log(0.2)) + idfs +  recurren
e
                ttot <- draw_tte(1,'lnorm',coef1=0.5, coef2=log(0.2)) + idfs +  recurrence
              }
            } else{ #If early metastatic
              start.early.mbc <- draw_tte(1,'lnorm',coef1=2.3, coef2=log(0.2))
              idfs <- ifelse(start.early.mbc<idfs,start.early.mbc,idfs)
              ttot.early <- min(ifelse(start.early.mbc<idfs,start.early.mbc,idfs),ttot.early)
              os.mbc <- draw_tte(1,'lnorm',coef1=0.8, coef2=log(0.2)) + start.early.mbc
              progression.mbc <- draw_tte(1,'lnorm',coef1=0.5, coef2=log(0.2)) + start.early.mbc
              ttot <- draw_tte(1,'lnorm',coef1=0.5, coef2=log(0.2)) + start.early.mbc
            }
            os <- min(os.mbc,os.early,nat.os.s)
          }) %>%
  add_tte(trt="noint",
          evts = c("start","ttot", "ttot.beva","progression.mbc", "os","idfs","ttot.early","remission","
ecurrence","start.early.mbc"),
          other_inp = c("os.early","os.mbc"),
          input={  #reference strategy
            start <- 0
            #Early
            idfs <- draw_tte(1,'lnorm',coef1=2, coef2=log(0.2),hr=1.2)
            ttot.early <- min(draw_tte(1,'lnorm',coef1=2, coef2=log(0.2),hr=1.2),idfs)
            os.early <- draw_tte(1,'lnorm',coef1=3, coef2=log(0.2),hr=1.2)

            #if patient has remission, check when will recurrence happen
            if (fl.remission) {
              recurrence <- idfs +draw_tte(1,'lnorm',coef1=2, coef2=log(0.2))
              remission <- idfs
              #if recurrence happens before death
              if (min(os.early,nat.os.s)>recurrence) {
                #Late metastatic (after finishing idfs and recurrence)
                os.mbc <- draw_tte(1,'lnorm',coef1=0.8, coef2=log(0.2)) + idfs  +  recurrence
                progression.mbc <- draw_tte(1,'lnorm',coef1=0.5, coef2=log(0.2)) + idfs +  recurren
e
                ttot <- draw_tte(1,'lnorm',coef1=0.5, coef2=log(0.2)) + idfs +  recurrence
              }
            } else{ #If early metastatic
              start.early.mbc <- draw_tte(1,'lnorm',coef1=2.3, coef2=log(0.2))
              idfs <- ifelse(start.early.mbc<idfs,start.early.mbc,idfs)
              ttot.early <- min(ifelse(start.early.mbc<idfs,start.early.mbc,idfs),ttot.early)
              os.mbc <- draw_tte(1,'lnorm',coef1=0.8, coef2=log(0.2)) + start.early.mbc
              progression.mbc <- draw_tte(1,'lnorm',coef1=0.5, coef2=log(0.2)) + start.early.mbc
              ttot <- draw_tte(1,'lnorm',coef1=0.5, coef2=log(0.2)) + start.early.mbc
            }
            os <- min(os.mbc,os.early,nat.os.s)
          })
```

添加事件反应

定义好事件的初始时间后，需要指定事件如何反应以及相互影响，需要使用evt_react_list对象和add_eactevt函数。这个函数需要指明受影响的事件和实际的影响（usually setting flags to 1 or 0, or creating new/adjusting events）。示例中添加一个服从Possion分布的不良事件，以及对特定事件的改（如通过从每个不良事件中减去1.5个月来修改死亡时间）。

可以使用以下对象帮助理解事件的发生：

Item	What does it do
curtime	Current event time (numeric)
prevtime	Time of the previous event (numeri)
cur_evtlist	Named vector of events that is yet to happen for that patient (named numeric vector)
evt	Current event being processed (character)
i	Patient being iterated (character)

simulation　　　　　　　　　　　　　　　　　　Simulation being iterated (numeri
)

建议将添加/修改的一类inputs/events作为一个list以group形式在函数中处理，而不是一个一个单独理。

添加事件（new_event）/修改事件（modify_event）时，必须定义或指明事件的名称和发生时间。

add_reactevt可使用的函数如下：

Function to use it	What does it do	Ho
modify_item() re events	Adds & Modifies items/flags/variables for fut modify_item(list("fl.idfs.ontx"=0,"fl.tx.beva"=0))	
new_event() ent	Adds events to the vector of events for that pat new_event(rep(list("ae"=curtime + 0.001),5))	
modify_event() me)	Modifies existing events by changing their t modify_event(list("os"=curtime +5, "ttot"=curtime+0.0001	

成本和效用（Costs and Utilities）

需注意，模型不考虑成本和效用运行是没有意义的。

效用

使用add_util函数定义效用。第一个参数指定事件，第二个指定治疗，第三个描述效用。

临时效用和循环效用可以使用相同的函数定义，但是循环效用需要指定开始时间和cycle length。

成本

使用add_cost函数定义成本。

运行模型

使用RunSim函数运行模型。需要指定模拟患者数、模拟次数、是否run a PSA等。

请注意，所选择的分布、事件数量以及事件之间的交互可能会对模型的运行时间产生重大影响。

```
#Logic is: per patient, per intervention, per event, react to that event.
results <- RunSim(
  npats=2000,                              # number of patients to be simulated
  n_sim=1,                                 # number of simulations to run
  psa_bool = FALSE,                        # use PSA or not. If n_sim > 1 and psa_bool = FALSE, then
ifference in outcomes is due to sampling (number of pats simulated)
```

```
  trt_list = c("int", "noint"),             # intervention list
  common_all_inputs = common_all_inputs,    # inputs common that do not change within a s
mulation
  common_pt_inputs = common_pt_inputs,      # inputs that change within a simulation but ar
 not affected by the intervention
  unique_pt_inputs = unique_pt_inputs,      # inputs that change within a simulation between i
terventions
  init_event_list = init_event_list,        # initial event list
  evt_react_list = evt_react_list,          # reaction of events
  util_ongoing_list = util_ongoing,
  cost_ongoing_list = cost_ongoing,
  cost_instant_list = cost_instant,
  ncores = 2,                               # number of cores to use, recommended not to use all
  drc = 0.035,                              # discount rate for costs
  drq = 0.035,                              # discount rate for QALYs
  input_out = c(                            # list of additional outputs (Flags, etc) that the user wants to
xport for each patient and event
                "os.early",
                "os.mbc",
                "nat.os.s",
                "sex_pt"
                )
          )
#> [1] "Simulation number: 1"
#> Warning in RunSim(npats = 2000, n_sim = 1, psa_bool = FALSE, trt_list =
#> c("int", : Item util_v is named. It is strongly advised to assign unnamed
#> objects if they are going to be processed in the model, as they can create
#> errors depending on how they are used within the model
#> Warning in RunSim(npats = 2000, n_sim = 1, psa_bool = FALSE, trt_list =
#> c("int", : Item cost_v is named. It is strongly advised to assign unnamed
#> objects if they are going to be processed in the model, as they can create
#> errors depending on how they are used within the model
#> [1] "Time to run iteration 1: 10.45s"
#> [1] "Total time to run: 10.45s"
```

模型输出

结果汇总

可以使用summary_results_det输出结果，summary_results_psa战士PSA结果，可以使用psa_ipd来绘图。

```
summary_results_det(results$final_output) #will print the last simulation!
#>            int      noint
#> costs 365317.90 267967.54
#> lys       12.99     11.83
#> qalys      9.82      8.97
#> ICER         NA  84062.52
#> ICUR         NA 114046.28

summary_results_psa(results$output_psa)
```

```
#>                    int              noint
#> costs 365318(365318, 365318) 267968(267968, 267968)
#> lys      12.99(12.99, 12.99)    11.83(11.83, 11.83)
#> qalys      9.82(9.82, 9.82)       8.97(8.97, 8.97)
#> ICER            NaN(NA, NA)    84063(84063, 84063)
#> ICUR            NaN(NA, NA) 114046(114046, 114046)
```

```
psa_ipd <- bind_rows(map(results$output_psa, "merged_df"))

psa_ipd[1:10,] %>%
  kable() %>%
  kable_styling(bootstrap_options = c("striped", "hover", "condensed", "responsive"))
```

evtname	evttime	cost	qaly	ly	pat_id	trt	sex_pt	at.os.s	os.early	os.mbc	total_costs	total_qalys	total_lys	s mulation
start	0.000000	0.0000	0.0000000	0.0000000	1	int	female	24.56429	17.0136	21.89380	414422.3	10.23653	12.87903	1
idfs	6.070305	328703.5455	4.1087943	5.4783924	1	int	female	24.56429	17.0136	21.89380	414422.3	10.23653	12.87903	1
ttot.early	6.070305	0.0000	0.0000000	0.0000000	1	int	female	24.56429	17.0136	21.89380	414422.3	10.23653	12.87903	1
remission	6.070305	0.0000	0.0000000	0.0000000	1	int	female	24.56429	17.0136	21.89380	414422.3	10.23653	12.87903	1
ttot.beva	8.579367	44868.1706	1.7557110	1.9507900	1	int	female	24.56429	17.0136	21.89380	414422.3	10.23653	12.87903	1
recurrence	13.987665	0.0000	3.3063563	3.6737292	1	int	female	24.56429	17.0136	21.89380	414422.3	10.23653	12.87903	1
os	17.013600	40850.6180	1.0656683	1.7761138	1	int	female	24.56429	17.0136	21.89380	414422.3	10.23653	12.87903	1
start	0.000000	0.0000	0.0000000	0.0000000	2	int	female	41.83077	19.2660	30.33222	379474.4	11.93604	14.02155	1
ttot.early	5.877182	319268.6445	3.9908581	5.3211441	2	int	female	41.83077	19.2660	30.33222	379474.4	11.93604	14.02155	1
ae	5.877282	816.9422	0.0000817	0.0000817	2	int	female	41.83077	19.2660	30.33222	379474.4	11.93604	14.02155	1

绘图

```
data_plot <- results$final_output$merged_df %>%
  # mutate(evttime = ifelse(evttime > 99, NA, evttime)) %>%
  filter(evtname != "start") %>%
  group_by(trt,evtname,simulation) %>%
  mutate(median = median(evttime)) %>%
  ungroup()

#Density
ggplot(data_plot) +
  geom_density(aes(fill = trt, x = evttime),
               alpha = 0.7) +
  geom_vline(aes(xintercept=median,col=trt)) +
  facet_wrap( ~ evtname, scales = "free_y") +
  scale_y_continuous(expand = c(0, 0)) +
  scale_x_continuous(expand = c(0, 0)) +
  theme_bw()
```

▯

▯

https://roche.github.io/descem/articles/example_ipd.html

```
ifelse(!"pacman" %in% installed.packages(),install.packages("pacman"),library(pacman))
p_load("descem","tidyverse","survival","survminer","flexsurv","flexsurvPlus","kableExtra")

#Define variables that do not change on any patient or intervention loop
common_all_inputs <- add_item(
  #Parameters from the survival models
  OS.scale = 3.1523,
  OS.shape = 1.1346,
  OS.coef.int = 0.3066, #Intervention effect

  # TTP.scale = as.numeric(TTP.fit$coef[2]),
  # TTP.shape = as.numeric(TTP.fit$coef[1]),
  # TTP.coef.int = as.numeric(TTP.fit$coef[3]), #Intervention effect

  TTP.scale = 0.5865,
  TTP.shape = 1.3757,
  TTP.coef.int = 1.0992, #Intervention effect

  #Utilities
  util.PFS = 0.6, #Utility while in progression-free state
  util.PPS = 0.4, #Utility while in progressed state
  disutil.PAE = -0.02, #One-off disutility of progression-accelerating event

  #Costs
  cost.drug.int = 85000, #Annual intervention cost
  cost.drug.ref = 29000, #Annual cost of reference treatment
  cost.admin.SC = 150, #Unit cost for each SC administration
  cost.admin.oral = 300, #One-off cost for oral administration
  cost.dm.PFS = 3000, #Annual disease-management cost in progression-free state
  cost.dm.PPS = 5000, #Annual disease-management cost in progressed state
  cost.ae.int = 2200, #Annual adverse event costs for intervention
  cost.ae.ref = 1400, #Annual adverse event costs for reference treatment
)


#Define variables that do not change as we loop through interventions for a patient
common_pt_inputs <- add_item(
  #Patient baseline characteristics
  Sex = rbinom(500,1,0.5), #Record sex of individual patient. 0 = Female; 1 =Male
  BLAge = rnorm(500,60,8), #Record patient age at baseline

  #Draw time to non-disease related death from a restricted Gompertz distribution
  nat.death = draw_resgompertz(1,shape=if(Sex == 1){0.102}else{0.115},
                               rate=if(Sex == 1){0.000016}else{0.0000041},
                               lower_bound = BLAge) # Baseline Age in years
)


#Define variables that change as we loop through treatments for each patient.
unique_pt_inputs <- add_item(
  fl.int  = 0, #Flag to determine if patient is on intervention. Initialized as 0, but will be change
 to current arm in the Start event.
  fl.prog = 0, #Flag to determine if patient has progressed. All patients start progression-free
  fl.ontx = 1, #Flag to determine if patient is on treatment. All patients start on treatment
  fl.PAE = 0,  #Flag to determine if progression-accelerating event occurred
  pfs.time = NA #Recording of time at progression
)


init_event_list <-
  #Events applicable to intervention
  add_tte(trt="int",
          evts = c("Start","TxDisc","Progression","PAE","Death"),
          input={
            Start <- 0
            Progression <- draw_tte(1,'weibull',coef1=TTP.shape, coef2= TTP.scale + TTP.coef.int,
eed = as.numeric(paste0(1,i,simulation)))
            TxDisc <- Inf #Treatment discontinuation will occur at progression
            Death <- min(draw_tte(1,'weibull',coef1=OS.shape, coef2= OS.scale + OS.coef.int, seed
= as.numeric(paste0(42,i,simulation))), nat.death) #Death occurs at earliest of disease-related
eath or non-disease-related death
            PAE <- draw_tte(1,'exp',coef1=-log(1-0.05)) #Occurrence of the progression-accelerati
g event has a 5% probability for the intervention arm
          }) %>%

  #Events applicable to reference treatment
  #Events are the same except that there will be no treatment discontinuation
  add_tte(trt="ref",
          evts = c("Start","Progression","PAE","Death"),
          input={
            Start <- 0
            Progression <- draw_tte(1,'weibull',coef1=TTP.shape, coef2= TTP.scale, seed = as.num
ric(paste0(1,i,simulation)))
            Death <- min(draw_tte(1,'weibull', coef1=OS.shape, coef2= OS.scale, seed = as.numeri
(paste0(42,i,simulation))), nat.death) #Death occurs at earliest of disease-related death or non
disease-related death
           PAE <- draw_tte(1,'exp',coef1=-log(1-0.15)) #Occurrence of the progression-accelerati
g event has a 15% probability for the reference arm
         })



evt_react_list <-
  add_reactevt(name_evt = "Start",
               input = {modify_item(list(fl.int = ifelse(trt=="int",1,0)))
               }) %>%
  add_reactevt(name_evt = "TxDisc",
               input = {modify_item(list("fl.ontx"= 0))
               }) %>%
  add_reactevt(name_evt = "Progression",
               input = {modify_item(list("pfs.time" = curtime, "fl.prog" = 1))
                        if(trt=="int"){modify_event(list("TxDisc" = curtime))} #Trigger treatment discont
nuation at progression
               }) %>%
  add_reactevt(name_evt = "Death",
               input = {modify_item(list("curtime"=Inf))
               }) %>%
  add_reactevt(name_evt = "PAE",
               input = {modify_item(list("fl.PAE"= 1))
                        #Event only accelerates progression if progression has not occurred yet
                        if(fl.prog == 0){modify_event(list("Progression" = max(draw_tte(1,'weibull',coef
=TTP.shape, coef2 = TTP.scale + TTP.coef.int*fl.int, hr = 1.2, seed = as.numeric(paste0(1,i,simu
ation))),curtime)))} #Occurrence of event accelerates progression by a factor of 1.2
               })



util_ongoing <- add_util(evt = c("Start","TxDisc","Progression","Death","PAE"),
                         trt = c("int", "ref"), #common utility across arms
                         util = ifelse(fl.prog == 0, util.PFS, util.PPS))

util_instant <- add_util(evt = c("PAE"),
                         trt = c("int", "ref"), #common utility across arms
                         util = disutil.PAE)




cost_ongoing <-
  #Drug costs are specific to each arm
  add_cost(
    evt = c("Start","TxDisc","Progression","Death","PAE"),
    trt = "int",
    cost = (cost.drug.int +
              cost.admin.SC * 12 + #Intervention is administered once a month
              cost.ae.int) * fl.ontx +
              ifelse(fl.prog == 0,cost.dm.PFS,cost.dm.PPS)) %>%
  add_cost(
    evt = c("Start","TxDisc","Progression","Death","PAE"),
    trt = "ref",
    cost = cost.drug.ref +
           cost.ae.ref + #No ongoing administration cost as reference treatment is oral
           ifelse(fl.prog == 0,cost.dm.PFS,cost.dm.PPS))

#One-off cost only applies to oral administration of reference treatment, applied at start of tr
atment
cost_instant <-  add_cost(
  evt = "Start",
  trt = "ref",
  cost = cost.admin.oral)


results <- RunSim(
  npats=2000,              # Simulating the number of patients for which we have IPD
  n_sim = 1,                                  # We run all patients once (per treatment)
  psa_bool = FALSE,                          # No PSA for this example
  trt_list = c("int", "ref"),
  common_all_inputs = common_all_inputs,
  common_pt_inputs = common_pt_inputs,
  unique_pt_inputs = unique_pt_inputs,
  init_event_list = init_event_list,
  evt_react_list = evt_react_list,
  util_ongoing_list = util_ongoing,
  cost_ongoing_list = cost_ongoing,
  cost_instant_list = cost_instant,
  ncores = 2,
  drc = 0.035,                              # discount rate for costs
  drq = 0.035,                              # discount rate for QALYs
  input_out = c("BLAge","Sex","nat.death","pfs.time")
)

summary_results_det(results$final_output)
```