Case Study 5: Emergency Room Discrete Event Simulation

Jonathan De Los Santos

The following case study models an emergency room with different system flows depending on the priority of incoming patients. First a distribution and its corresponding parameters must be decided from a historical dataset. Afterwards, we will move to modeling the ER simulation and proposing improvements to the process.

Case:

Patients arrive at the Emergency Room following an unknown probability distribution (stream 1). They will be treated by either of two doctors.

A proportion of the patients are classified as NIA (need immediate attention) and the rest as CW (can wait). NIA patients are given the highest priority, 3, see a doctor as soon as possible for 40 ± 30 minutes (stream 2), then have their priority reduced to 2 and wait until a doctor is free again, when they receive further treatment for 30 ± 20 minutes (stream 3) and are discharged.

CW patients initially receive a priority of 1 and are treated (when their turn comes) for 15 ± 10 minutes (stream 4); their priority is then increased to 2, they wait again until a doctor is free, receive 10 ± 5 minutes (stream 5) of final treatment and are discharged.

An important aspect of this system is that patients who have already seen the doctor once compete with newly arriving patients who need a doctor. As indicated, patients who have already seen the doctor once have a priority level of 2 (either increased from 1 to 2 or decreased from 3 to 2). Thus, there is one shared queue for the first treatment activity and the final treatment activity. In addition, we assume that the doctors are interchangeable. That is, it does not matter which of the two doctors performs the first or final treatment.

Simulate for 20 days of continuous operation, 24 hours per day. Note, the inter-arrival time and type of 100 patients are collected (based on a randomly selected weekday data). The data is attached.

1. *Analyze your results and explain your suggestions for reducing the waiting time of the patients.*
2. *What is the average flow-time for NIA and CW patients before or after applying suggestions. different suggestions.*

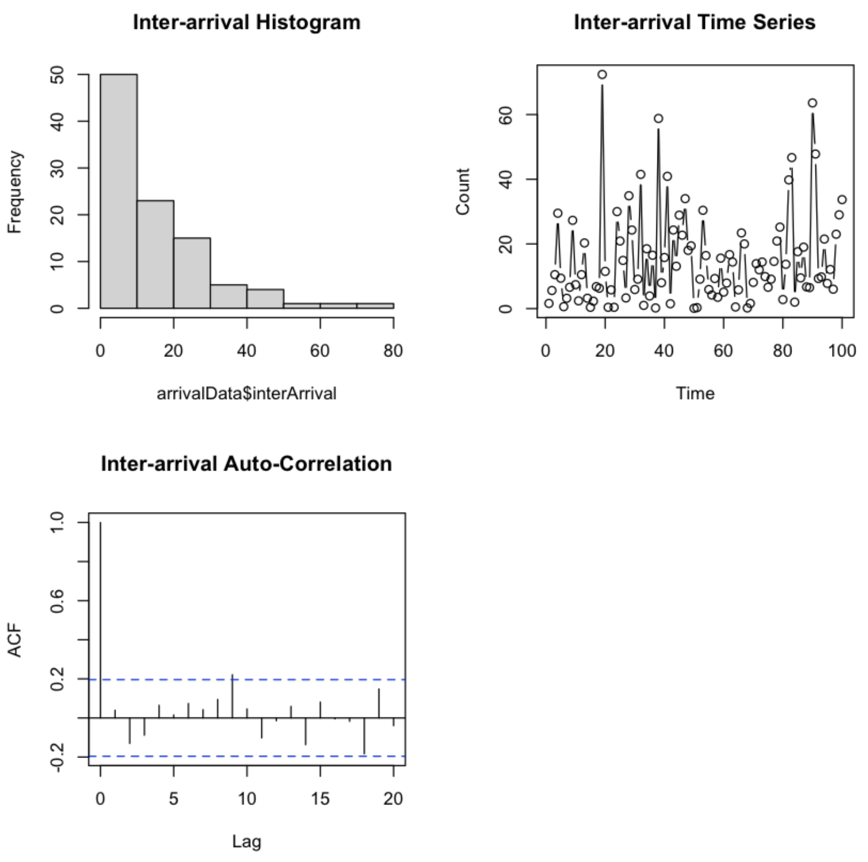
3. *Discuss the utilization of doctors before or after applying suggestions.*

Data Input Analysis

Before starting our discrete event simulation, we need to examine our patient arrival data and make a determination about its probability distribution.

In [1]:
```r
arrivalData = read.csv("Data Sets/GroupCase5Data.csv")
```

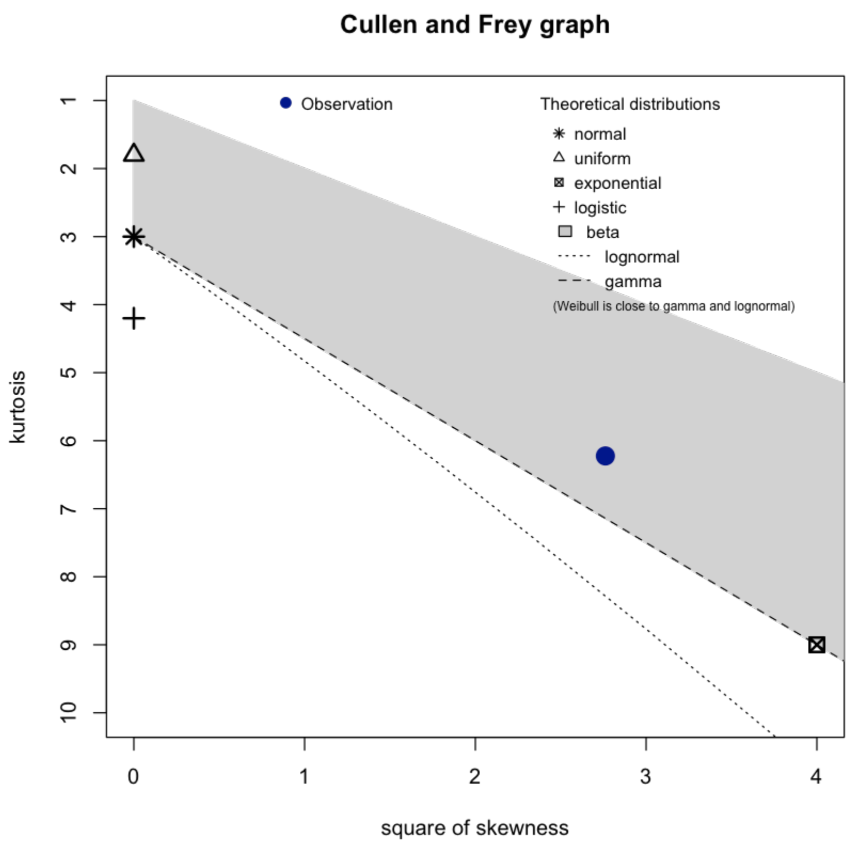
In [10]:
```r
suppressWarnings(library(gridExtra))
suppressWarnings(library(simmer.plot))

# Set up a 2x2 plot
par(mfrow = c(2, 2))

hist(arrivalData$interArrival, main="Inter-arrival Histogram")
plot(arrivalData$interArrival, type="b", main="Inter-arrival Time Series", y
acf(arrivalData$interArrival, main= "Inter-arrival Auto-Correlation")

# Reset the layout to default
par(mfrow = c(1, 1))
```



We can examine the data using the `fitdistrplus` library. In this instance, we will assume our arrival data is continuuous (`discrete = FALSE`).

In [12]:
```
suppressWarnings(library(fitdistrplus))
descdist(arrivalData$interArrival, discrete = FALSE)
```

```
summary statistics
------
min:  0.1   max:  72.4
median:  10.2
mean:  15.077
estimated sd:  14.35904
estimated skewness:  1.661798
estimated kurtosis:  6.224021
```



Based on our Cullen and Frey graph, we will examine lognormal, gamma, and exponential distributions.

In [15]:
```
fit.exp = fitdist(arrivalData$interArrival, "exp")
fit.gamma = fitdist(arrivalData$interArrival, "gamma")
fit.lnorm = fitdist(arrivalData$interArrival, "lnorm")
```

In [16]:
```
summary(fit.exp)
gofstat(fit.exp)
summary(fit.gamma)
gofstat(fit.gamma)
summary(fit.lnorm)
gofstat(fit.lnorm)
```

```
Fitting of the distribution ' exp ' by maximum likelihood
Parameters :
       estimate  Std. Error
rate 0.06632619 0.006631111
Loglikelihood:  -371.317   AIC:  744.6341   BIC:  747.2393
Goodness-of-fit statistics
                              1-mle-exp
Kolmogorov-Smirnov statistic 0.07025042
Cramer-von Mises statistic   0.04728725
Anderson-Darling statistic   0.40070355

Goodness-of-fit criteria
                               1-mle-exp
Akaike's Information Criterion  744.6341
Bayesian Information Criterion  747.2393
Fitting of the distribution ' gamma ' by maximum likelihood
Parameters :
        estimate Std. Error
shape 0.96354630 0.11955644
rate  0.06388923 0.01025346
Loglikelihood:  -371.2718   AIC:  746.5435   BIC:  751.7539
Correlation matrix:
          shape      rate
shape 1.0000000 0.7728234
rate  0.7728234 1.0000000
Goodness-of-fit statistics
                             1-mle-gamma
Kolmogorov-Smirnov statistic  0.07804184
Cramer-von Mises statistic    0.05957576
Anderson-Darling statistic    0.41550172

Goodness-of-fit criteria
                               1-mle-gamma
Akaike's Information Criterion    746.5435
Bayesian Information Criterion    751.7539
Fitting of the distribution ' lnorm ' by maximum likelihood
Parameters :
        estimate Std. Error
meanlog 2.111460 0.13654252
sdlog   1.365425 0.09654991
Loglikelihood:  -384.1864   AIC:  772.3729   BIC:  777.5832
Correlation matrix:
        meanlog sdlog
meanlog       1     0
sdlog         0     1
```

```
Goodness-of-fit statistics
                             1-mle-lnorm
Kolmogorov-Smirnov statistic   0.1479491
Cramer-von Mises statistic     0.4586617
Anderson-Darling statistic     2.8735691

Goodness-of-fit criteria
                               1-mle-lnorm
Akaike's Information Criterion    772.3729
Bayesian Information Criterion    777.5832
```

By comparing the Akaike's and Bayesion Information Criteria as well as the Loglikelihoods, it appears the data is best fit by an exponential distribution.

The rate (λ) for our exponential distribution is ~0.0663 which represents how many arrivals per minute we can expect at the ER. **The recipocral of this parameter ($\frac{1}{λ}$) gives us the estimated inter-arrival time of 15.1 minutes.**

Now we need to determine the probabilities of each patient classification from the data set so we can set up our trajectories accordingly.

In [26]:
```
# Count the number of NIA and CW patients
nia_count <- sum(arrivalData$type == "NIA")
cw_count <- sum(arrivalData$type == "CW")

# Calculate probabilities
total_patients <- nrow(arrivalData)
nia_prob <- nia_count / total_patients
cw_prob <- cw_count / total_patients

cat("NIA Patient Probability: ", nia_prob, "\n")
cat("CW Patient Probability: ", cw_prob)
```

```
NIA Patient Probability:  0.18
CW Patient Probability:  0.82
```

Initial DES Model

In [83]:
```
suppressWarnings(library(simmer))

set.seed(123)

nia_prob <- 0.25
cw_prob <- 0.75

envs <- lapply(1:20, function(i) {
  env <- simmer("ER") %>%
    add_resource("doctor", 2)

  patient <- trajectory("Patient Path") %>%
```

```r
    branch(function()
      sample(1:2, size=1, replace=TRUE, prob=c(nia_prob,cw_prob)), continue=

      # NIA Patient
      trajectory("Needs Immediate Assistance") %>%
        set_attribute("priority", 3) %>%
        set_attribute("patientType", -3) %>%
        set_prioritization(c(3, 3, T)) %>%

        # NIA sees doctor for 10-70 minutes
        seize("doctor", 1) %>%
        timeout(function() runif(1, 10, 70)) %>%
        release("doctor", 1) %>%

        # Priority is reduced
        set_attribute("priority", 2) %>%
        set_prioritization(c(2, 3, T)) %>%

        # Received further treatment for 10-50 minutes
        seize("doctor", 1) %>%
        timeout(function() runif(1, 10, 50)) %>%
        release("doctor", 1),

      # CW Patient
      trajectory("Can Wait") %>%
        set_attribute("priority", 1) %>%
        set_attribute("patientType", -1) %>%
        set_prioritization(c(1, 3, T)) %>%

        # CW sees doctor for 5-25 minutes
        seize("doctor", 1) %>%
        timeout(function() runif(1, 5, 25)) %>%
        release("doctor", 1) %>%

        # Priority is raised
        set_attribute("priority", 2) %>%
        set_prioritization(c(2, 3, T)) %>%

        # Received further treatment for 5-15 minutes
        seize("doctor", 1) %>%
        timeout(function() runif(1, 5, 15)) %>%
        release("doctor", 1)
    )
  env %>%
    add_generator("patient", patient, function() rexp(1,1/15), mon = 2)

  env %>%
    # Simulate 24 hour day in minutes
    run(1440)
})
```

Initial Model Results

The initial model with its two doctors reveals long flow and wait times for both NIA and CW patients. CW patients feel the brunt of this pain, with nearly twice as long of a flow time that is mostly spent waiting.

Looking at our plots for waiting and flow time, we see that these both continuosly increase over time showing that the system perpetually backs up. This is especially evident in our resource usage graph, which shows over all simulations that both doctors remain at max utilization after about halfway through the day.

In [115…

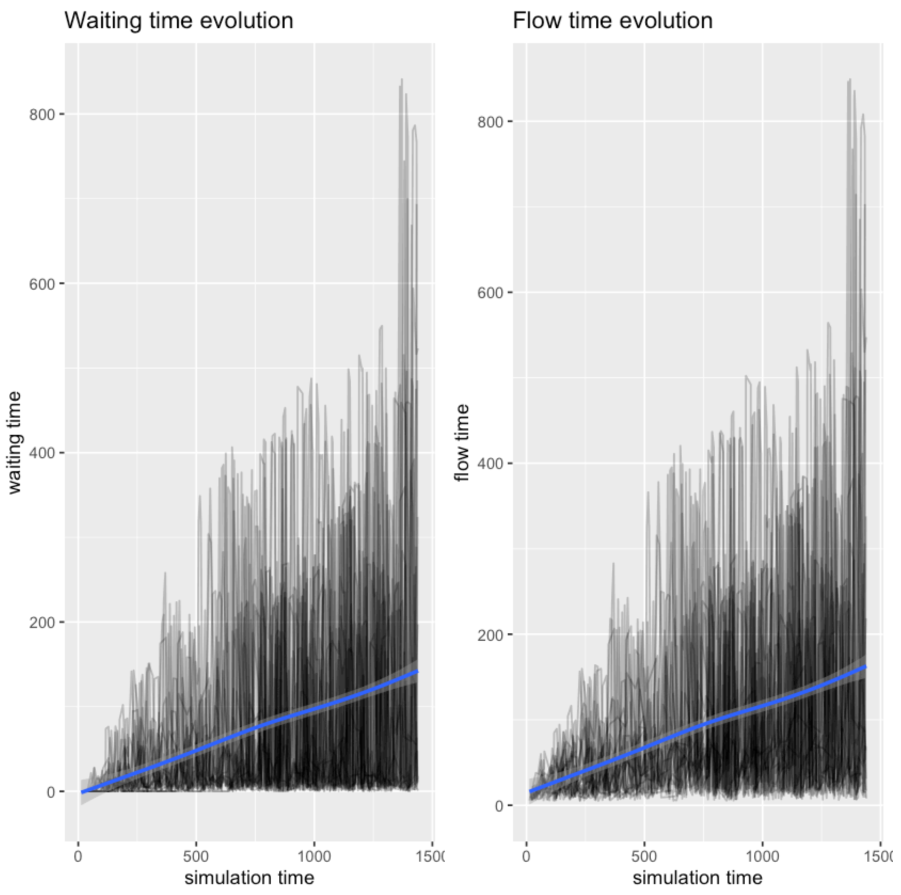
```r
x1 <- get_mon_arrivals(envs)
x2 <- get_mon_attributes(envs)
all <- merge(x1, x2, by=c("name", "replication"), all = T)
all <- na.omit(all)

# High Priority flow and wait time
NIA <- subset(all, all$value == 3)
nia.flowTime = (NIA$end_time-NIA$start_time)
nia.waitTime = (NIA$end_time - NIA$start_time) - NIA$activity_time

# Low Priority flow and wait time
CW <- subset(all, all$value == 1)
cw.flowTime = (CW$end_time-CW$start_time)
cw.waitTime = (CW$end_time - CW$start_time) - CW$activity_time

cat("NIA Priority Average Flow Time: ", mean(nia.flowTime, na.rm = T), "\n")
cat("NIA Priority Average Wait Time: ", mean(nia.waitTime, na.rm = T), "\n")
cat("CW Priority Average Flow Time: ", mean(cw.flowTime, na.rm = T), "\n")
cat("CW Priority Average Wait Time: ", mean(cw.waitTime, na.rm = T))
```

```
NIA Priority Average Flow Time:  108.1191
NIA Priority Average Wait Time:  37.8005
CW Priority Average Flow Time:  206.1234
CW Priority Average Wait Time:  181.075
```

In [104…

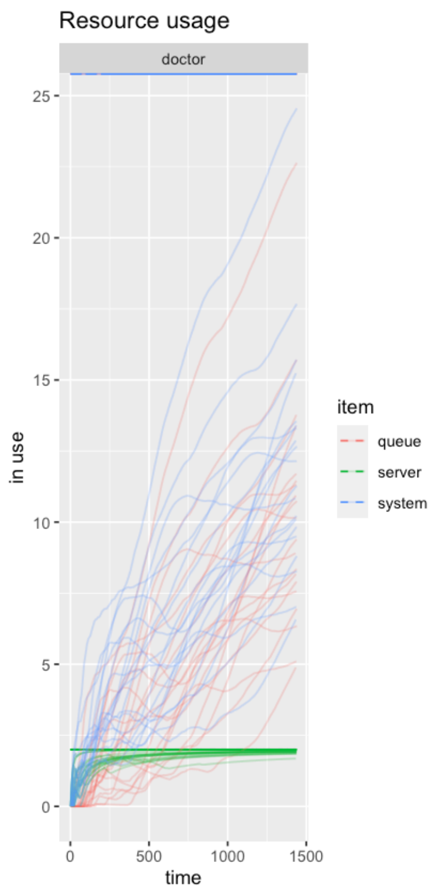
```r
suppressWarnings(library(gridExtra))

arrivals <- get_mon_arrivals(envs, per_resource = T)

p1 <- plot(arrivals, metric = "waiting_time")
p2 <- plot(arrivals, metric = "flow_time")

grid.arrange(p1,p2, ncol=2)
```

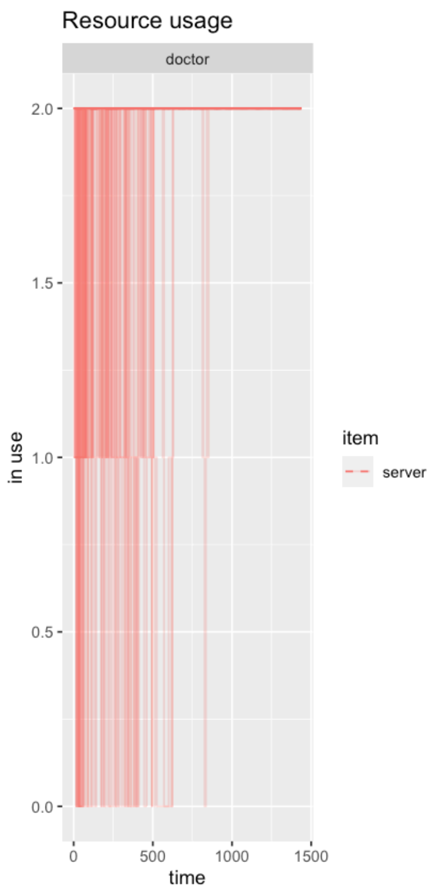
```
`geom_smooth()` using method = 'gam' and formula 'y ~ s(x, bs = "cs")'

`geom_smooth()` using method = 'gam' and formula 'y ~ s(x, bs = "cs")'
```



In [105…

```
resources <- get_mon_resources(envs)
p3 <- plot(resources, metric = "usage")
p4 <- plot(get_mon_resources(envs), metric = "usage", "doctor", items = "ser

grid.arrange(p3,p4, ncol=2)
```





DES Iteration 1: Add a Doctor

First we will attempt an obvious solution, adding the additional resource of one doctor.

In [113…

```
suppressWarnings(library(simmer))

set.seed(123)

nia_prob <- 0.25
cw_prob <- 0.75

envs2 <- lapply(1:20, function(i) {
  env2 <- simmer("ER") %>%
    add_resource("doctor", 3)

  patient <- trajectory("Patient Path") %>%

    branch(function()
      sample(1:2, size=1, replace=TRUE, prob=c(nia_prob,cw_prob)), continue=

      # NIA Patient
      trajectory("Needs Immediate Assistance") %>%
        set_attribute("priority", 3) %>%
        set_attribute("patientType", -3) %>%
        set_prioritization(c(3, 3, T)) %>%
        # NIA sees doctor for 10-70 minutes
        seize("doctor", 1) %>%
        timeout(function() runif(1, 10, 70)) %>%
        release("doctor", 1) %>%

        # Priority is reduced
        set_attribute("priority", 2) %>%
        set_prioritization(c(2, 3, T)) %>%

        # Received further treatment for 10-50 minutes
        seize("doctor", 1) %>%
        timeout(function() runif(1, 10, 50)) %>%
        release("doctor", 1),

      # CW Patient
      trajectory("Can Wait") %>%
        set_attribute("priority", 1) %>%
        set_attribute("patientType", -1) %>%
        set_prioritization(c(1, 3, T)) %>%

        # CW sees doctor for 5-25 minutes
        seize("doctor", 1) %>%
        timeout(function() runif(1, 5, 25)) %>%
        release("doctor", 1) %>%

        # Priority is raised
        set_attribute("priority", 2) %>%
        set_prioritization(c(2, 3, T)) %>%

        # Received further treatment for 5-15 minutes
        seize("doctor", 1) %>%
        timeout(function() runif(1, 5, 15)) %>%
        release("doctor", 1)
    )
  env2 %>%
    add_generator("patient", patient, function() rexp(1,1/15), mon = 2)

  env2 %>%
    # Simulate 24 hour day in minutes
    run(1440)
})
```

Iteration 1 Results

Adding a single doctor dramatically improved flow and wait times for both NIA and CW patients. NIA patients still have higher flow times due to their longer doctor visits, but their wait time is now a very small part of their overall activity time. CW patients still spend about half their time waiting, but their flow time is now actually less than NIA patients.

The waiting and flow time plots show that they are no longer constantly increasing throughout the day. After an initial spike there is a dip in both times, although it is unclear if the trajectory would continue upwards after a single day. The resource usage plots do seem to indicate that usage continues to decrease through the end of the day, and the doctors no longer remain at constant max utilization.

In [114…

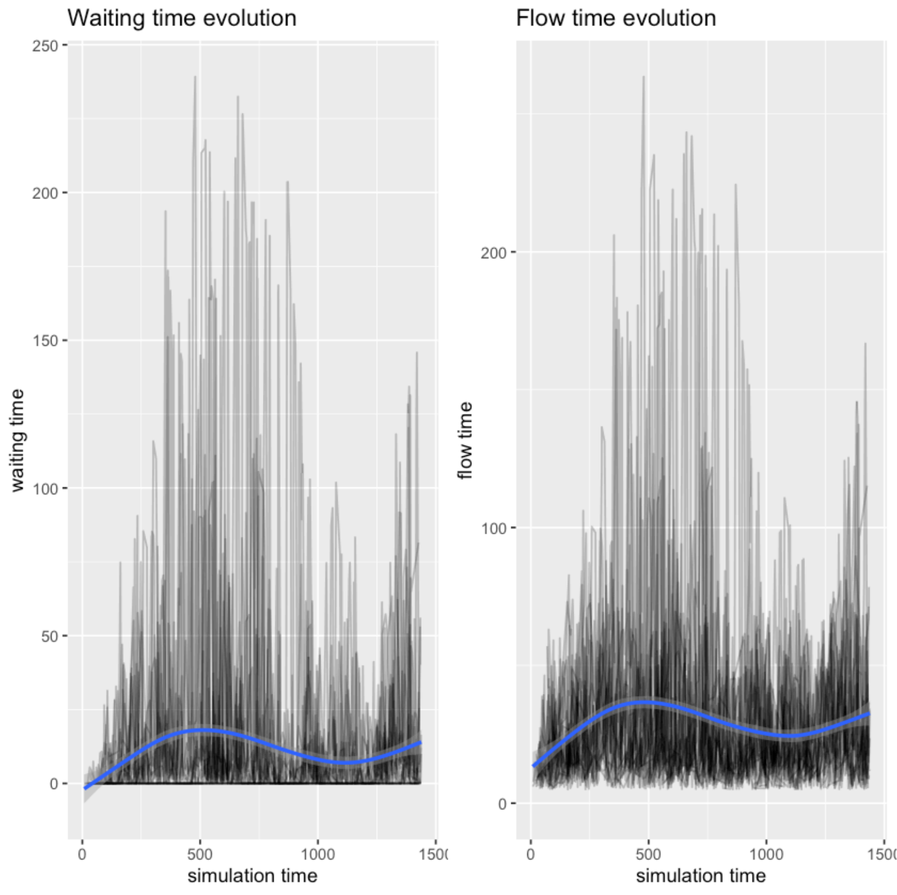
```r
x1 <- get_mon_arrivals(envs2)
x2 <- get_mon_attributes(envs2)
all <- merge(x1, x2, by=c("name", "replication"), all = T)
all <- na.omit(all)

# High Priority flow and wait time
NIA <- subset(all, all$value == 3)
nia.flowTime = (NIA$end_time-NIA$start_time)
nia.waitTime = (NIA$end_time - NIA$start_time) - NIA$activity_time

# Low Priority flow and wait time
CW <- subset(all, all$value == 1)
cw.flowTime = (CW$end_time-CW$start_time)
cw.waitTime = (CW$end_time - CW$start_time) - CW$activity_time

cat("NIA Priority Average Flow Time: ", mean(nia.flowTime, na.rm = T), "\n")
cat("NIA Priority Average Wait Time: ", mean(nia.waitTime, na.rm = T), "\n")
cat("CW Priority Average Flow Time: ", mean(cw.flowTime, na.rm = T), "\n")
cat("CW Priority Average Wait Time: ", mean(cw.waitTime, na.rm = T))
```

```
NIA Priority Average Flow Time:  77.03679
NIA Priority Average Wait Time:  8.458684
CW Priority Average Flow Time:  52.24723
CW Priority Average Wait Time:  27.27617
```

In [108…

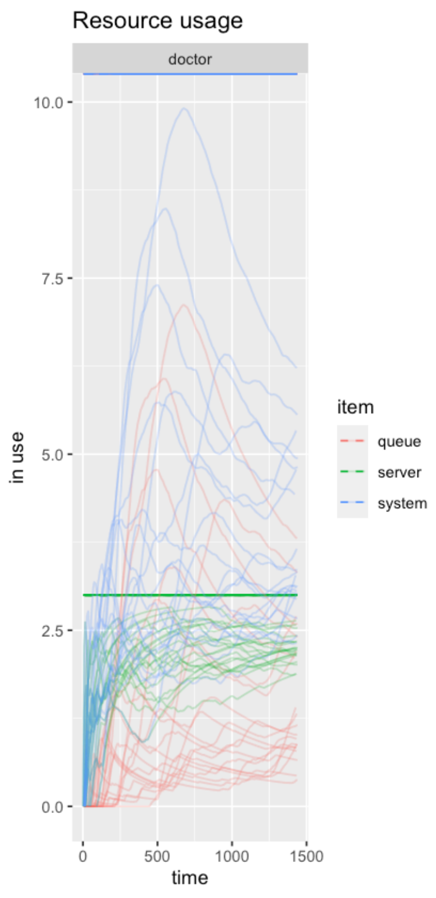
```r
suppressWarnings(library(gridExtra))

arrivals <- get_mon_arrivals(envs2, per_resource = T)

p5 <- plot(arrivals, metric = "waiting_time")
p6 <- plot(arrivals, metric = "flow_time")

grid.arrange(p5,p6, ncol=2)
```

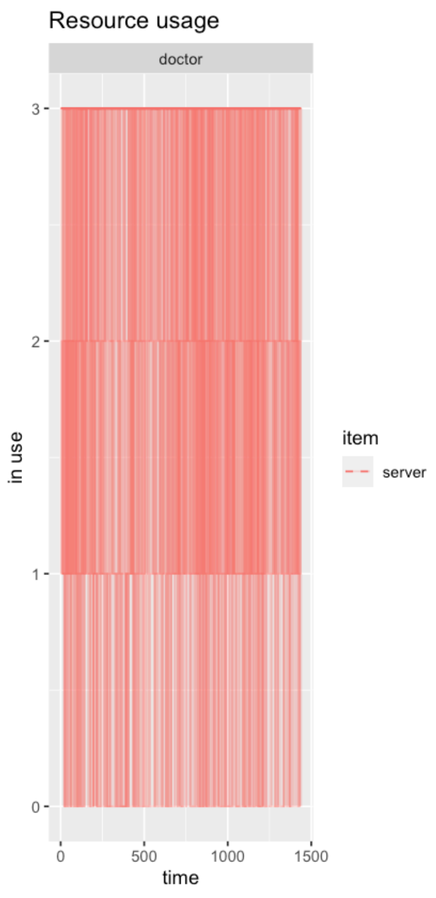
```
`geom_smooth()` using method = 'gam' and formula 'y ~ s(x, bs = "cs")'

`geom_smooth()` using method = 'gam' and formula 'y ~ s(x, bs = "cs")'
```



In [110…

```
resources <- get_mon_resources(envs2)
p7 <- plot(resources, metric = "usage")
p8 <- plot(get_mon_resources(envs2), metric = "usage", "doctor", items = "se

grid.arrange(p7,p8, ncol=2)
```





DES Iteration 2: Doctors Grow on Trees

One element of this model that sticks out is the dramatically higher seize times of doctors by NIA patients. While improvements to flow and wait times could be improved by reducing those times, I prefer that hospitals increase resources rather than reduce the quality of care. Hospital shareholders hate this one weird trick, but they can build their own simulation models.

This iteration adds an additional doctor resource. It's not very creative, but it is quite effective.

In [120…

```r
suppressWarnings(library(simmer))

set.seed(123)

nia_prob <- 0.25
cw_prob <- 0.75

envs3 <- lapply(1:20, function(i) {
  env3 <- simmer("ER") %>%
    add_resource("doctor", 4)

  patient <- trajectory("Patient Path") %>%

    branch(function()
      sample(1:2, size=1, replace=TRUE, prob=c(nia_prob,cw_prob)), continue=

      # NIA Patient
      trajectory("Needs Immediate Assistance") %>%
        set_attribute("priority", 3) %>%
        set_attribute("patientType", -3) %>%
        set_prioritization(c(3, 3, T)) %>%

        # NIA sees doctor for 10-70 minutes
        seize("doctor", 1) %>%
        timeout(function() runif(1, 10, 70)) %>%
        release("doctor", 1) %>%

        # Priority is reduced
        set_attribute("priority", 2) %>%
        set_prioritization(c(2, 3, T)) %>%

        # Received further treatment for 10-50 minutes
        seize("doctor", 1) %>%
        timeout(function() runif(1, 10, 50)) %>%
        release("doctor", 1),

      # CW Patient
      trajectory("Can Wait") %>%
        set_attribute("priority", 1) %>%
        set_attribute("patientType", -1) %>%
        set_prioritization(c(1, 3, T)) %>%

        # CW sees doctor for 5-25 minutes
        seize("doctor", 1) %>%
        timeout(function() runif(1, 5, 25)) %>%
        release("doctor", 1) %>%

        # Priority is raised
        set_attribute("priority", 2) %>%
        set_prioritization(c(2, 3, T)) %>%

        # Received further treatment for 5-15 minutes
        seize("doctor", 1) %>%
        timeout(function() runif(1, 5, 15)) %>%
        release("doctor", 1)
    )
  env3 %>%
    add_generator("patient", patient, function() rexp(1,1/15), mon = 2)

  env3 %>%
    # Simulate 24 hour day in minutes
    run(1440)
})
```

Iteration 2 results

The low priority (CW) patients are the winners of adding a fourth doctor as their flow time is nearly halved. However, wait times are now extremely quick for all patients and not increasing over time. The resource usage graphs are now more populated around the middle of the plots rather than the top, suggesting that doctors spend less time fully utilized.

Perhaps this is not ideal in business school terms, but for an ER I would suggest it is beneficial to invest in lower wait times and not optimize solely for max utilization of physicians.

In [119…

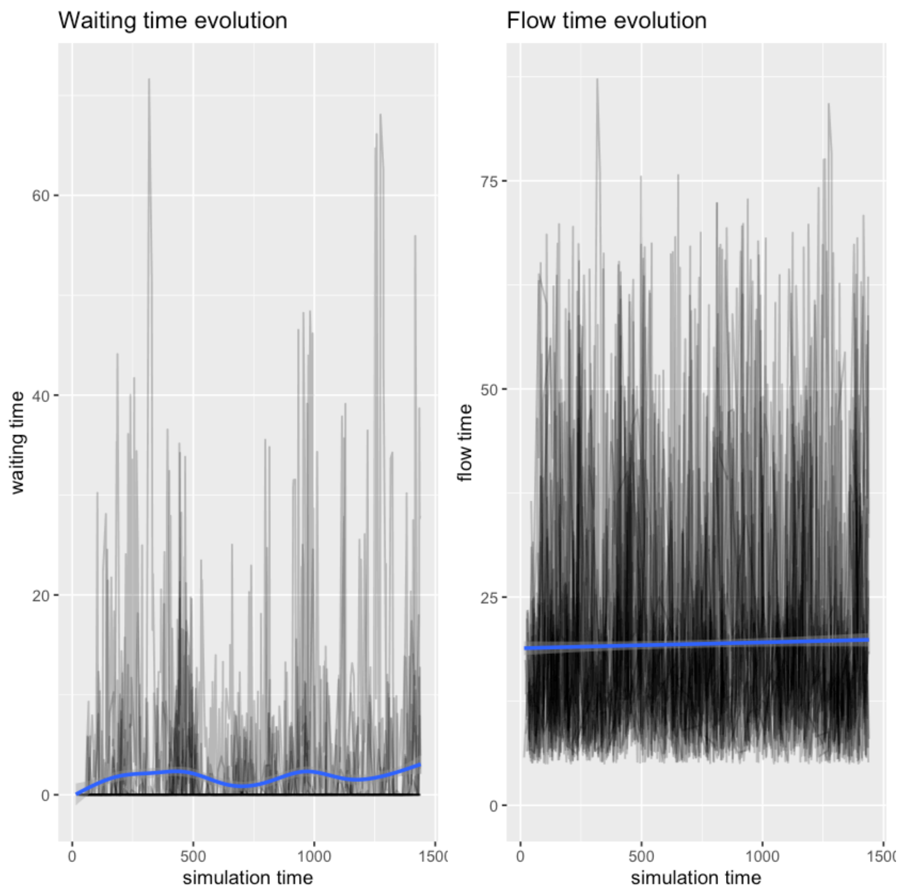
```
x1 <- get_mon_arrivals(envs3)
x2 <- get_mon_attributes(envs3)
all <- merge(x1, x2, by=c("name", "replication"), all = T)
all <- na.omit(all)

# High Priority flow and wait time
NIA <- subset(all, all$value == 3)
nia.flowTime = (NIA$end_time-NIA$start_time)
nia.waitTime = (NIA$end_time - NIA$start_time) - NIA$activity_time

# Low Priority flow and wait time
CW <- subset(all, all$value == 1)
cw.flowTime = (CW$end_time-CW$start_time)
cw.waitTime = (CW$end_time - CW$start_time) - CW$activity_time

cat("NIA Priority Average Flow Time: ", mean(nia.flowTime, na.rm = T), "\n")
cat("NIA Priority Average Wait Time: ", mean(nia.waitTime, na.rm = T), "\n")
cat("CW Priority Average Flow Time: ", mean(cw.flowTime, na.rm = T), "\n")
cat("CW Priority Average Wait Time: ", mean(cw.waitTime, na.rm = T))
```

```
NIA Priority Average Flow Time:  71.74587
NIA Priority Average Wait Time:  2.122879
CW Priority Average Flow Time:  28.76042
CW Priority Average Wait Time:  3.914827
```

In [121…

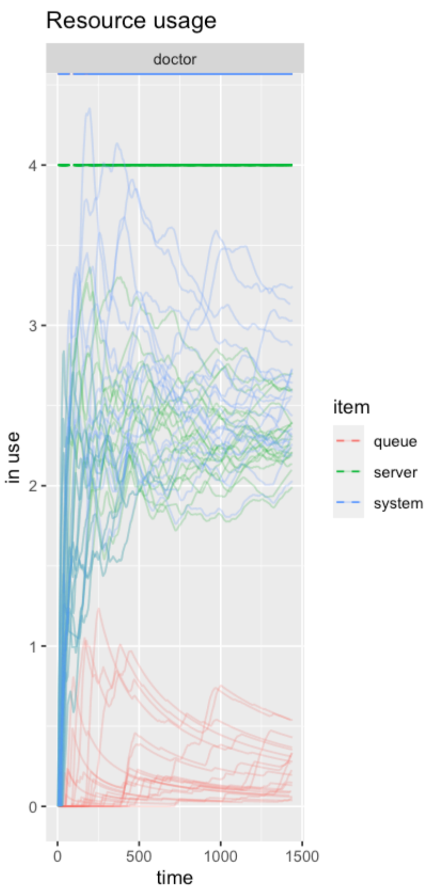
```
suppressWarnings(library(gridExtra))

arrivals <- get_mon_arrivals(envs3, per_resource = T)

p9 <- plot(arrivals, metric = "waiting_time")
p10 <- plot(arrivals, metric = "flow_time")

grid.arrange(p9,p10, ncol=2)
```

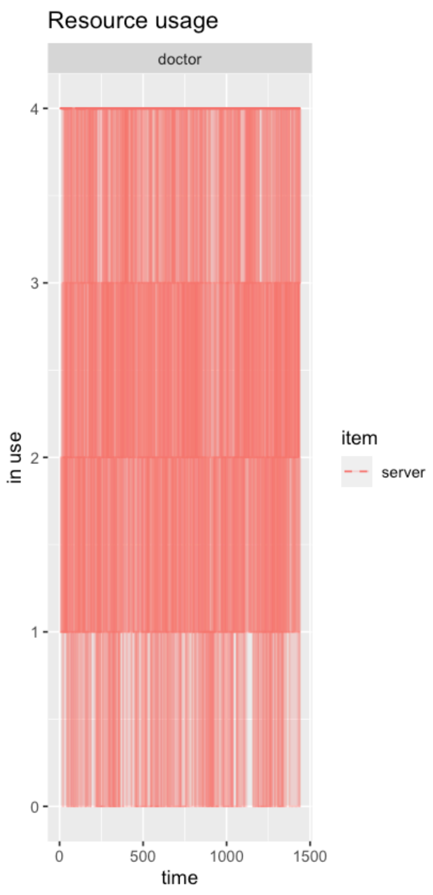
```
`geom_smooth()` using method = 'gam' and formula 'y ~ s(x, bs = "cs")'

`geom_smooth()` using method = 'gam' and formula 'y ~ s(x, bs = "cs")'
```



In [123…

```
resources <- get_mon_resources(envs3)
p11 <- plot(resources, metric = "usage")
p12 <- plot(get_mon_resources(envs3), metric = "usage", "doctor", items = "s

grid.arrange(p11,p12, ncol=2)
```


Resource usage
doctor
in use
time
item
queue
server
system
0
1
2
3
4
0
500
1000
1500


Resource usage
doctor
in use
time
item
server
0
1
2
3
4
0
500
1000
1500