The Effect of Voicemails on Voter Interaction with a Political Phone Bank by Simulation

Biostatistics 213: Final Project

Chad Pickering

12/14/2018

Introduction.

Political organizations need to be wary about how they spend their limited resources, particularly the time spent contacting voters by phone. At a phone banking operation, it is of interest to know if the time spent leaving voicemails that prompt voters to call back and pledge support is worth the time. Will there be a sufficient increase in voters who call back as a result of this time investment that more voters are spoken to per hour on average? If so, we need to know what proportion of unanswered calls have to call back to make up the net loss of outgoing calls that would be sacrificed at the expense of leaving voicemails. To understand this, two competing simulations are devised, one where unanswered calls are left voicemails and one where they are not.

Methods.

In both versions of the simulation (voicemails left and not), we assume that no two outgoing calls are made simultaneously; that is, no two outgoing calls overlap (overcoming this assumption is a worthwhile extension). We also assume that phone number look-up is done automatically via software so search time is negligible. Another assumption is that no call goes directly to voicemail; there is no distinction between cell phones and landlines. Each outgoing call has a 10% chance of being answered immediately upon the first attempt, simulated with a Bernoulli random variable. When a voter answers the call, we assume a negligible ring time, a talk time distributed exponential with parameter $\lambda = 100$ (a mean uninterrupted talk time of $\frac{1}{100}$ of an hour, or 36 seconds, probably on the short side), and a constant "buffer" time of $\frac{1}{120}$ of an hour (30 seconds) appended to the end of each call for note-taking on the part of the employee. This buffer time is set as a constant at the beginning of the simulation and its use changes depending on the context. In reality, the buffer time could have been simulated as a normal random variable with mean $\frac{1}{120}$ of an hour and a very low variance, but with several hundred repetitions, the variance introduced would not change the outcome of either simulation in a significant way. For those individuals who answer the call on this first attempt, they have a probability of zero of calling the organization back in either scenario, whether or not a voicemail is left.

If the individual does not answer and no voicemail is left, only a buffer time of 30 seconds, now representing the total ring time, is added. If a voicemail is left, we simply add two buffer times for a total time of 60 seconds, one for the ring time and one representing the time to record the voicemail. To determine if the individual will call back or not, we simulate a beta random variable to represent their personal probability of callback, which is then inputted as the probability of success in a Bernoulli random variable. This binary outcome determines whether or not a callback by this specific individual is made in the future. In the scenario where voicemails are not left, the beta random variable has a default mean of 0.05 (this will vary later in the analysis), whereas when voicemails are left, the default mean is 0.25 (this is also varied later). The callback time itself is also determined, distributed as a gamma random variable with a shape parameter of 1 and a rate parameter of 2 (this particular parameterization is equivalent to an exponential random variable with $\lambda = 2$, but other gamma parameterizations can be chosen). This means that, on average, an individual who calls back does so about 30 minutes after the first call is attempted. The choice of the gamma distribution allows for more flexibility of center of location, variance, and right tail skewness. These callback times could have also been distributed uniformly with very little substantive change in the coming analysis, but the absence of a defined right bound of the random variable cannot be determined before the simulation is terminated so this is impossible to achieve.

At the end of each iteration, we check to see if any callbacks had occurred since the last voter was contacted. We assume that any incoming calls in this short time period are put on hold in the order in which they arrive, and are taken in that order as soon as the current outgoing call is completed. The talk time of incoming calls are also distributed exponential with parameter $\lambda = 100$, and an additional 30 second buffer time for note taking is added as well. When an incoming call is completed, its callback time that is stored in the first entry associated with that voter ID is removed from consideration so that the next soonest incoming call can be found.

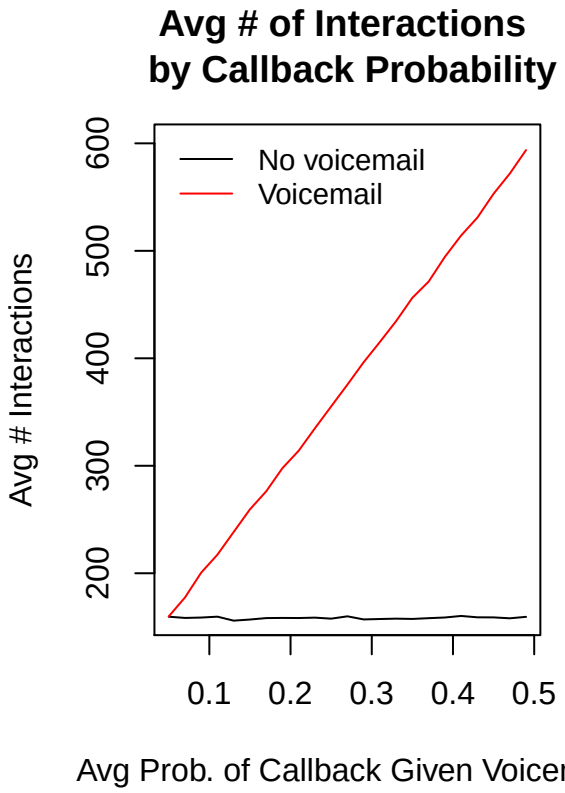
When we make the 1000th call that goes unanswered, the simulation completes that iteration, which could include assigning a callback time for the individual, and terminates. This stopping point is arbitrary and is another argument in the wrapper function that can be amended. Termination time is determined by number of unanswered calls rather than a user-set time in hours in order to prevent a situation where the average number of calls taken per hour is small and the set termination time is not sufficiently large enough to generate a substantial number of total calls.

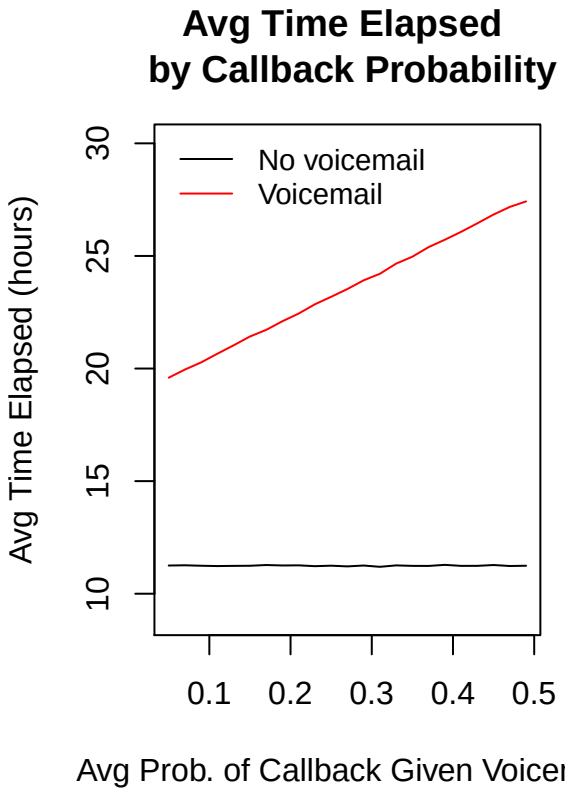
The default settings are: 150 total repetitions; 30 second buffer time; 1000 calls not answered before simulation terminates; 10% chance of a voter answering a call on the first attempt; 36 second average uninterrupted call time; 5% chance of voter calling back in no voicemail scenario; 25% chance of voter calling back in voicemail scenario. Unless otherwise stated, the defaults are always adhered to.

Analysis.

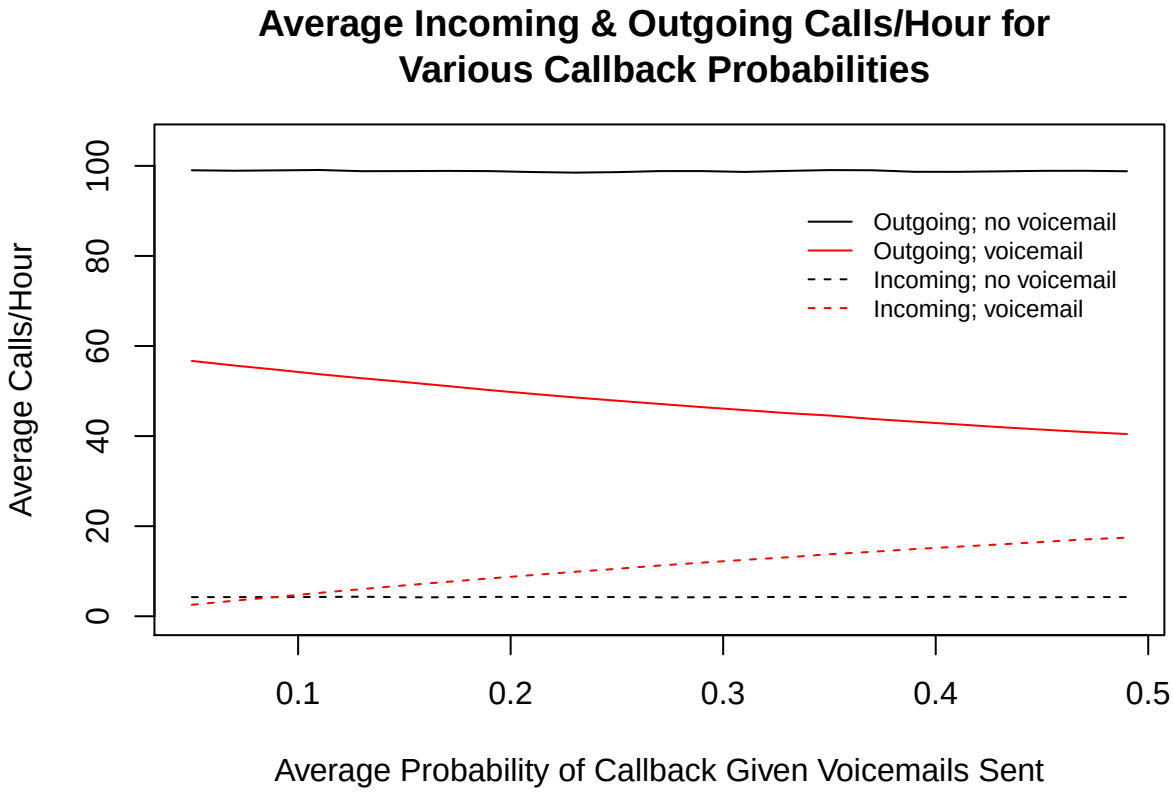
First, we should check to see if some expected patterns hold to confirm that the simulation is functioning as it should. For example, the average number of interactions (e.g. total number of voters spoken to) holds at about 158 when all defaults are kept constant and no voicemails are left. When voicemails are left, the number of contacts made increases by about 10 voters per additional 1% probability of callback. As expected, as probability of callback increases, more voters are interacted with.

Further, we expect that in a scenario where voicemails are left, we spend a longer time until the 1000th unanswered call than if voicemails are not left. With default parameters, the average time spent until the 1000th unanswered call is complete is about 11.25 hours, whereas when leaving voicemails, we expect to spend about 19.6 hours until the 1000th unanswered call when the probability of callback is also 5%. It is expected that about 11 additional minutes are spent for every additional 1% probability of callback.



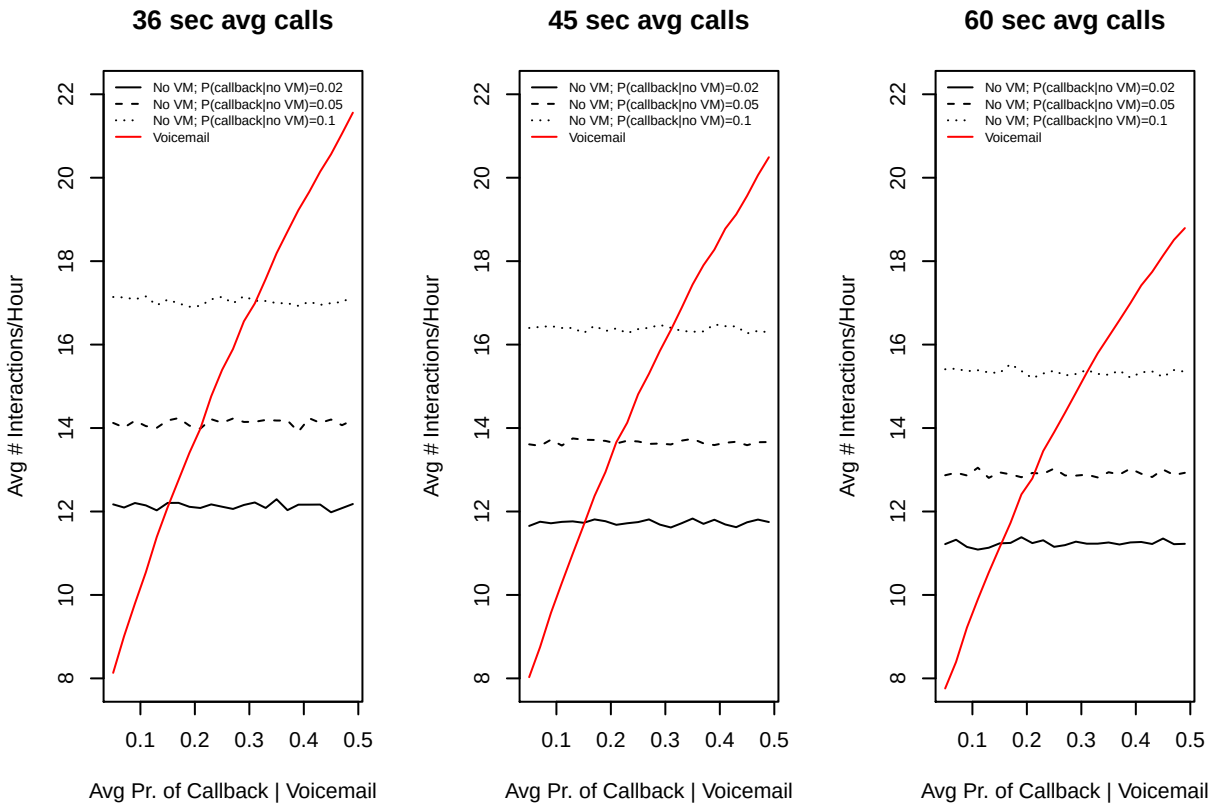


For the scenario where voicemails are not left, we expect to have about 98.8 outgoing calls per hour and about 4.3 incoming calls per hour with all defaults. In the case of voicemails, we expect that there be about 56.7 outgoing calls per hour and 2.5 incoming calls per hour with a 5% chance of callback. As the plot below shows, the outgoing calls per hour decrease at a decreasing rate and the incoming calls per hour increase at a decreasing rate as the probability of callback increases. The ongoing question is: at what point is the rate of incoming calls large enough to overtake the loss of potential interactions via outgoing calls?



When calls are assumed to be distributed exponential with $\lambda = 100$ (36 second average call time) and no voicemails are left, we expect about 12.1 interactions per hour when the probability of callback is 2%, 14.1 when the callback probability is 5%, and about 17 when it's 10%. In order for voicemails to be a worthwhile investment, the phone banking operation must at least break even with interactions per hour when voicemails are administered. According to the simulation, the callback probability using voicemails must be at least 15% if it is known that the callback probability without voicemails is 2% to interact with more voters per hour on average, a more than seven-fold increase. Additionally, we see that the callback probability using voicemails must be at least 21%, a more than four-fold increase, to interact with more voters if the callback probability without voicemails is 5%. If the callback probability without voicemails is 10%, we must see at least a 31% callback rate with voicemails implemented, about a three-fold increase. Analogous analysis is used for the other call time distributions.

Even if calls are on average one minute long ($\lambda = 60$), the callback probability using voicemails still must be at least about 15% if we know that the callback probability without voicemails is 2% to interact with more voters per hour on average. It appears that, at the very least, in the range of reasonable average call time lengths explored here, break even points do not change significantly for each respective callback probability. The lack of precision of the simulation allows only a cursory look at the differences between results per choice of average call time; results shown here are not certain to follow if extrapolation to call time averages outside this range are desired.



Conclusion.

Regardless of average call time in the range explored, recording voicemails should only be done if they increase callback probability by a fairly wide margin, particularly if the original callback probability without voicemail implementation is in the low single digits. Future additions to this simulation should include the implementation of more than one caller; multiple calls should be able to be placed, as well as taken, at once in the organization.

Code Appendix.

```
load("~/Documents/Syncplicity/ChadSync/BIOSTATISTICS/Fall 2018/213/213_finalproj_env.RData")
call_process <- function(num_sim=150, # constant
                         t.buffer=1/120, # constant
                         n.no_answer_threshold=1000, # constant
                         p.call_answered=0.1, # constant
                         avg_calls_per_hour=100, # constant
                         avg_p_callback.no_vm=1/20, # constant
                         avg_p_callback.vm=1/4) # vary later
{

  ## NO VOICEMAIL SCENARIO:

  # Initiate statistics to be recorded in each repetition:
  no.answer_immed <- 0
  no.callbacks <- 0
  contacts.made <- 0
  no.ids <- 0
```

```
no.calls <- 0
time_el <- 0
avg_outgoing_per_hr <- 0
avg_incoming_per_hr <- 0
avg_intxn_per_hr <- 0

for(i in 1:num_sim){

  # initialize time at 0
  t <- 0
  # vector of length n.calls of times at which calls are completed
  t.complete <- 0
  # vector of length n.calls that records if a call was answered or not
  answer_status <- 0
  # vector of length n.calls that records the probability of a callback
  p_callback <- 0
  # vector of length n.calls that records whether the person called back
  callback_status <- 0
  # vector of length n.calls that records the time of callback
  t.callback <- 0
  # number of calls made through time t
  n.calls <- 0
  # number of callbacks received through time t
  n.callbacks <- 0
  # number of calls answered through time t
  n.answer <- 0
  # number of calls not answered through time t
  n.no_answer <- 0
  # keep track of ID on the contact list
  contact.id <- 0

  # while n unanswered calls have not been achieved:
  while(n.no_answer<n.no_answer_threshold){

    # Call is made; probability p of answering, 1-p of not answering:
    if(rbinom(1, 1, p.call_answered)==1){ # if call is answered:
      # time at which call is complete - add talk time and note time
      t <- t + rexp(1, avg_calls_per_hour) + t.buffer
      n.calls <- n.calls + 1 # increment number of calls
      contact.id[n.calls] <- n.calls-n.callbacks # assign contact.id
      n.answer <- n.answer + 1 # increment number of answered calls
      t.complete[n.calls] <- t # store current time
      answer_status[n.calls] <- 1 # store last call status
      p_callback[n.calls] <- 0 # no chance of callback
      # no chance of callback - all 0
      callback_status[n.calls] <- rbinom(1, 1, p_callback[n.calls])
      t.callback[n.calls] <- NA
    } else { # if call is not answered (voicemail not left):
      t <- t + t.buffer # time at which call is complete - add ring time
      n.calls <- n.calls + 1 # increment number of calls
      contact.id[n.calls] <- n.calls-n.callbacks # assign contact.id
      n.no_answer <- n.no_answer + 1 # increment number of unanswered calls
      t.complete[n.calls] <- t # store current time
```

```
      answer_status[n.calls] <- 0 # store last call status
      # probability of callback has default mean 1/20
      p_callback[n.calls] <- rbeta(1, 1, (1/avg_p_callback.no_vm)-1)
      # determine if person called back
      callback_status[n.calls] <- rbinom(1, 1, p_callback[n.calls])
      # If callback occurs, determine time of callback:
      if(callback_status[n.calls]==1){
        t.callback[n.calls] <- t + rgamma(1, 1, 2) # mean of 30 min delay
      } else { # if callback does not occur, assign NA to time of callback
        t.callback[n.calls] <- NA
      }
    }

    # After each call is completed, check to see if, during the duration
    # of that call, there was an incoming callback. If so, take the call next.
    # Do not stop this process until the condition is false;
    # otherwise, some callbacks adjacent to others will be ignored.
    while(min(t.callback, na.rm=TRUE)<=t.complete[n.calls] &&
          length(min(t.callback, na.rm=TRUE))>0 && n.calls>1){
      n.calls <- n.calls + 1 # increment number of calls
      n.callbacks <- n.callbacks + 1 # increment number of callbacks completed
      # set contact.id to id who is calling
      contact.id[n.calls] <- contact.id[which(t.callback==min(t.callback, na.rm=TRUE))]
      # add time talking to person who called back + buffer time for notes
      t <- t + rexp(1, avg_calls_per_hour) + t.buffer
      # assign new time to t.complete in new row entry
      t.complete[n.calls] <- t
      # contact with person is made, make answer_status 1
      answer_status[n.calls] <- 1
      p_callback[n.calls] <- 0 # irrelevant, make 0
      callback_status[n.calls] <- 0 # irrevelant, make 0
      t.callback[n.calls] <- NA # irrelevant, make NA to match the others
      # now replace t.callback for that contact.id to NA
      t.callback[which(t.callback==min(t.callback, na.rm=TRUE))] <- NA
    }

  }

  # Create next entries in vectors that will be averaged:
  contacts.made_no.vm <- sum(answer_status)
  total.time_no.vm <- t.complete[length(t.complete)]
  no.answer_immed[i] <- n.answer
  no.callbacks[i] <- n.callbacks
  contacts.made[i] <- contacts.made_no.vm
  no.ids[i] <- max(contact.id)
  no.calls[i] <- n.calls
  time_el[i] <- t
  avg_outgoing_per_hr[i] <- (n.calls-n.callbacks)/total.time_no.vm
  avg_incoming_per_hr[i] <- n.callbacks/total.time_no.vm
  avg_intxn_per_hr[i] <- contacts.made_no.vm/total.time_no.vm

}
```

```r
# Create first data frame:
statistic.no_vm <- c(round(mean(no.answer_immed), 3),  round(mean(contacts.made), 3),
                     round(mean(no.ids), 3), round(mean(no.callbacks), 3), round(mean(no.calls), 3),
                     round(mean(time_el), 3), round(mean(avg_outgoing_per_hr), 3),
                     round(mean(avg_incoming_per_hr), 3), round(mean(avg_intxn_per_hr), 3))
df.no_vm <- data.frame(statistic.no_vm)
colnames(df.no_vm) <- NULL
row.names(df.no_vm) <- NULL


## VOICEMAIL SCENARIO:

# Initiate statistics to be recorded in each repetition:
no.answer_immed <- 0
no.callbacks <- 0
contacts.made <- 0
no.ids <- 0
no.calls <- 0
time_el <- 0
avg_outgoing_per_hr <- 0
avg_incoming_per_hr <- 0
avg_intxn_per_hr <- 0

for(i in 1:num_sim){

  # initialize time at 0
  t <- 0
  # vector of length n.calls of times at which calls are completed
  t.complete <- 0
  # vector of length n.calls that records if a call was answered or not
  answer_status <- 0
  # vector of length n.calls that records the probability of a callback
  p_callback <- 0
  # vector of length n.calls that records whether the person called back
  callback_status <- 0
  # vector of length n.calls that records the time of callback
  t.callback <- 0
  # number of calls made through time t
  n.calls <- 0
  # number of callbacks through time t
  n.callbacks <- 0
  # number of calls answered through time t
  n.answer <- 0
  # number of calls not answered through time t
  n.no_answer <- 0
  # keep track of ID on the contact list
  contact.id <- 0

  # while n unanswered calls have not been achieved:
  while(n.no_answer<n.no_answer_threshold){

    # Call is made; probability p of answering, 1-p of not answering:
    if(rbinom(1, 1, p.call_answered)==1){ # call is answered
```

```
    # time at which call is complete - add talk time and note time
    t <- t + rexp(1, avg_calls_per_hour) + t.buffer
    n.calls <- n.calls + 1 # increment number of calls
    contact.id[n.calls] <- n.calls-n.callbacks # assign contact.id
    n.answer <- n.answer + 1 # increment number of answered calls
    t.complete[n.calls] <- t # store current time
    answer_status[n.calls] <- 1 # store last call status
    p_callback[n.calls] <- 0 # no chance of callback
    # no chance of callback - all 0
    callback_status[n.calls] <- rbinom(1, 1, p_callback[n.calls])
  } else { # call is not answered and voicemail is left
    # time at which call is complete - add ring time + voice mail time
    t <- t + (2*t.buffer)
    n.calls <- n.calls + 1 # increment number of calls
    contact.id[n.calls] <- n.calls-n.callbacks # assign contact.id
    n.no_answer <- n.no_answer + 1 # increment number of no answers
    t.complete[n.calls] <- t # store current time
    answer_status[n.calls] <- 0 # store last call status
    # probability of callback has default mean 1/4
    p_callback[n.calls] <- rbeta(1, 1, (1/avg_p_callback.vm)-1)
    # determine if person called back
    callback_status[n.calls] <- rbinom(1, 1, p_callback[n.calls])
    # If callback occurs, determine time of callback:
    if(callback_status[n.calls]==1){
      t.callback[n.calls] <- t + rgamma(1, 1, 2) # mean of 30 min delay
    } else { # callback does not occur, assign NA to time of callback
      t.callback[n.calls] <- NA
    }
  }

  # After each call is completed, check to see if, during the duration
  # of that call, there was an incoming callback. If so, take the call next.
  # Do not stop this process until the condition is false;
  # otherwise, some callbacks adjacent to others will be ignored.
  while(min(t.callback, na.rm=TRUE)<=t.complete[n.calls] &&
        length(min(t.callback, na.rm=TRUE))>0 && n.calls>1){
    n.calls <- n.calls + 1 # increment number of calls
    n.callbacks <- n.callbacks + 1 # increment number of callbacks completed
    # set contact.id to id who is calling
    contact.id[n.calls] <- contact.id[which(t.callback==min(t.callback, na.rm=TRUE))]
    # add time talking to person who called back + buffer time for notes
    t <- t + rexp(1, avg_calls_per_hour) + t.buffer
    # assign new time to t.complete in new row entry
    t.complete[n.calls] <- t
    # contact with person is made, make answer_status 1
    answer_status[n.calls] <- 1
    p_callback[n.calls] <- 0 # irrelevant, make 0
    callback_status[n.calls] <- 0 # irrevelant, make 0
    t.callback[n.calls] <- NA # irrelevant, make NA to match the others
    # now replace t.callback for that contact.id to NA
    t.callback[which(t.callback==min(t.callback, na.rm=TRUE))] <- NA
  }
```

```
    }

    # Create next entries in vectors that will be averaged:
    contacts.made_no.vm <- sum(answer_status)
    total.time_no.vm <- t.complete[length(t.complete)]
    no.answer_immed[i] <- n.answer
    no.callbacks[i] <- n.callbacks
    contacts.made[i] <- contacts.made_no.vm
    no.ids[i] <- max(contact.id)
    no.calls[i] <- n.calls
    time_el[i] <- t
    avg_outgoing_per_hr[i] <- (n.calls-n.callbacks)/total.time_no.vm
    avg_incoming_per_hr[i] <- n.callbacks/total.time_no.vm
    avg_intxn_per_hr[i] <- contacts.made_no.vm/total.time_no.vm

  }

  # Complete construction of data frame:
  labels <- c("Avg. # Calls Answered Immediately",  "Avg. # Contacts Made",
              "Avg. # Unique IDs", "Avg. # Callbacks",
              "Avg. Total # Calls", "Time Elapsed",
              "Avg. Outgoing Calls/Hour", "Avg. Incoming Calls/Hour",
              "Avg. # Interactions/Hour")
  statistic.vm <- c(mean(no.answer_immed), mean(contacts.made), mean(no.ids),
                    mean(no.callbacks), mean(no.calls),
                    round(mean(time_el), 3), round(mean(avg_outgoing_per_hr), 3),
                    round(mean(avg_incoming_per_hr), 3), round(mean(avg_intxn_per_hr), 3))
  df.vm <- data.frame(statistic.vm)
  full_df <- cbind(df.no_vm, df.vm)
  colnames(full_df) <- NULL
  row.names(full_df) <- labels
  full_df <- as.data.frame(full_df)
  full_df

}
# Vector of callback probabilities to iterate over:
vector_probs <- seq(0.05, 0.49, by=0.02)

n_contacts_no_vm <- c()
n_contacts_vm <- c()
time_el_no_vm <- c()
time_el_vm <- c()
outgoing_no_vm <- c()
outgoing_vm <- c()
incoming_no_vm <- c()
incoming_vm <- c()

for(i in 1:length(vector_probs)){
  n_contacts_no_vm[i] <- call_process(avg_p_callback.vm = vector_probs[i])[2, 1][[1]]
  n_contacts_vm[i] <- call_process(avg_p_callback.vm = vector_probs[i])[2, 2][[1]]
  time_el_no_vm[i] <- call_process(avg_p_callback.vm = vector_probs[i])[6, 1][[1]]
  time_el_vm[i] <- call_process(avg_p_callback.vm = vector_probs[i])[6, 2][[1]]
  outgoing_no_vm[i] <- call_process(avg_p_callback.vm = vector_probs[i])[7, 1][[1]]
```

```
  outgoing_vm[i] <- call_process(avg_p_callback.vm = vector_probs[i])[7, 2][[1]]
  incoming_no_vm[i] <- call_process(avg_p_callback.vm = vector_probs[i])[8, 1][[1]]
  incoming_vm[i] <- call_process(avg_p_callback.vm = vector_probs[i])[8, 2][[1]]
}
### Avg # Contacts Made

par(mfrow=c(1, 2))

plot(vector_probs, n_contacts_no_vm, type="l",
     main="Avg # of Interactions \n by Callback Probability",
     xlab="Avg Prob. of Callback Given Voicemail",
     ylab="Avg # Interactions", lty=1, ylim = c(160, 600))
lines(vector_probs, n_contacts_vm, type = "l", lty=1, col="red")
legend('topleft', c('No voicemail', 'Voicemail'),
       col = c("black", "red"), lty = c(1, 1), bty = 'n', cex = 0.9)

plot(vector_probs, time_el_no_vm, type="l",
     main="Avg Time Elapsed \n by Callback Probability",
     xlab="Avg Prob. of Callback Given Voicemail",
     ylab="Avg Time Elapsed (hours)", lty=1, ylim = c(9, 30))
lines(vector_probs, time_el_vm, type = "l", lty=1, col="red")
legend('topleft', c('No voicemail', 'Voicemail'),
       col = c("black", "red"), lty = c(1, 1), bty = 'n', cex = 0.9)
### Avg # Outgoing Calls/Hour vs Average # Incoming Calls/Hour

par(mfrow=c(1, 1))

plot(vector_probs, incoming_no_vm, type="l",
     main="Average Incoming & Outgoing Calls/Hour for \n Various Callback Probabilities",
     xlab="Average Probability of Callback Given Voicemails Sent",
     ylab="Average Calls/Hour",
     ylim = c(0, 105), lty=2)
lines(vector_probs, incoming_vm, type = "l", lty=2, col="red")
lines(vector_probs, outgoing_no_vm, type = "l", lty=1, col="black")
lines(vector_probs, outgoing_vm, type = "l", lty=1, col="red")
legend(x=0.33, y=94, c('Outgoing; no voicemail', 'Outgoing; voicemail',
                       'Incoming; no voicemail', 'Incoming; voicemail'),
       col = c("black", "red", "black", "red"), lty = c(1, 1, 2, 2),
       bty = 'n', cex = 0.75)
### Avg Interactions/Hour for P(callback|v.m.) = {0.05,...,0.5}

# Lambda = 100

no_vm_list_0.02 <- c()
vm_list_0.02 <- c()

for(i in 1:length(vector_probs)){
  no_vm_list_0.02[i] <- call_process(avg_p_callback.no_vm = 0.02,
                                     avg_p_callback.vm = vector_probs[i])[9, 1][[1]]
  vm_list_0.02[i] <- call_process(avg_p_callback.no_vm = 0.02,
                                  avg_p_callback.vm = vector_probs[i])[9, 2][[1]]
}
```

```r
final_df_0.02 <- data.frame(cbind(vector_probs, no_vm_list_0.02, vm_list_0.02))
colnames(final_df_0.02) <- c("avg_prob_callback", "avg_intxns_no_vm", "avg_intxns_vm")

no_vm_list_0.05 <- c()
vm_list_0.05 <- c()

for(i in 1:length(vector_probs)){
  no_vm_list_0.05[i] <- call_process(avg_p_callback.vm = vector_probs[i])[9, 1][[1]]
  vm_list_0.05[i] <- call_process(avg_p_callback.vm = vector_probs[i])[9, 2][[1]]
}

final_df_0.05 <- data.frame(cbind(vector_probs, no_vm_list_0.05, vm_list_0.05))
colnames(final_df_0.05) <- c("avg_prob_callback", "avg_intxns_no_vm", "avg_intxns_vm")

no_vm_list_0.1 <- c()
vm_list_0.1 <- c()

for(i in 1:length(vector_probs)){
  no_vm_list_0.1[i] <- call_process(avg_p_callback.no_vm = 0.1,
                                    avg_p_callback.vm = vector_probs[i])[9, 1][[1]]
  vm_list_0.1[i] <- call_process(avg_p_callback.no_vm = 0.1,
                                 avg_p_callback.vm = vector_probs[i])[9, 2][[1]]
}

final_df_0.1 <- data.frame(cbind(vector_probs, no_vm_list_0.1, vm_list_0.1))
colnames(final_df_0.1) <- c("avg_prob_callback", "avg_intxns_no_vm", "avg_intxns_vm")

# Lambda = 80

no_vm_list_0.02_80 <- c()
vm_list_0.02_80 <- c()

for(i in 1:length(vector_probs)){
  no_vm_list_0.02_80[i] <- call_process(avg_p_callback.no_vm = 0.02, avg_calls_per_hour = 80,
                                        avg_p_callback.vm = vector_probs[i])[9, 1][[1]]
  vm_list_0.02_80[i] <- call_process(avg_p_callback.no_vm = 0.02, avg_calls_per_hour = 80,
                                     avg_p_callback.vm = vector_probs[i])[9, 2][[1]]
}

final_df_0.02_80 <- data.frame(cbind(vector_probs, no_vm_list_0.02_80, vm_list_0.02_80))
colnames(final_df_0.02_80) <- c("avg_prob_callback", "avg_intxns_no_vm", "avg_intxns_vm")

no_vm_list_0.05_80 <- c()
vm_list_0.05_80 <- c()

for(i in 1:length(vector_probs)){
  no_vm_list_0.05_80[i] <- call_process(avg_p_callback.vm = vector_probs[i],
                                        avg_calls_per_hour = 80)[9, 1][[1]]
  vm_list_0.05_80[i] <- call_process(avg_p_callback.vm = vector_probs[i],
                                     avg_calls_per_hour = 80)[9, 2][[1]]
}

final_df_0.05_80 <- data.frame(cbind(vector_probs, no_vm_list_0.05_80, vm_list_0.05_80))
```

```r
colnames(final_df_0.05_80) <- c("avg_prob_callback", "avg_intxns_no_vm", "avg_intxns_vm")

no_vm_list_0.1_80 <- c()
vm_list_0.1_80 <- c()

for(i in 1:length(vector_probs)){
  no_vm_list_0.1_80[i] <- call_process(avg_p_callback.no_vm = 0.1,
                                       avg_p_callback.vm = vector_probs[i],
                                       avg_calls_per_hour = 80)[9, 1][[1]]
  vm_list_0.1_80[i] <- call_process(avg_p_callback.no_vm = 0.1,
                                    avg_p_callback.vm = vector_probs[i],
                                    avg_calls_per_hour = 80)[9, 2][[1]]
}

final_df_0.1_80 <- data.frame(cbind(vector_probs, no_vm_list_0.1_80, vm_list_0.1_80))
colnames(final_df_0.1_80) <- c("avg_prob_callback", "avg_intxns_no_vm", "avg_intxns_vm")

# Lambda = 60

no_vm_list_0.02_60 <- c()
vm_list_0.02_60 <- c()

for(i in 1:length(vector_probs)){
  no_vm_list_0.02_60[i] <- call_process(avg_p_callback.no_vm = 0.02, avg_calls_per_hour = 60,
                                        avg_p_callback.vm = vector_probs[i])[9, 1][[1]]
  vm_list_0.02_60[i] <- call_process(avg_p_callback.no_vm = 0.02, avg_calls_per_hour = 60,
                                     avg_p_callback.vm = vector_probs[i])[9, 2][[1]]
}

final_df_0.02_60 <- data.frame(cbind(vector_probs, no_vm_list_0.02_60, vm_list_0.02_60))
colnames(final_df_0.02_60) <- c("avg_prob_callback", "avg_intxns_no_vm", "avg_intxns_vm")

no_vm_list_0.05_60 <- c()
vm_list_0.05_60 <- c()

for(i in 1:length(vector_probs)){
  no_vm_list_0.05_60[i] <- call_process(avg_p_callback.vm = vector_probs[i],
                                        avg_calls_per_hour = 60)[9, 1][[1]]
  vm_list_0.05_60[i] <- call_process(avg_p_callback.vm = vector_probs[i],
                                     avg_calls_per_hour = 60)[9, 2][[1]]
}

final_df_0.05_60 <- data.frame(cbind(vector_probs, no_vm_list_0.05_60, vm_list_0.05_60))
colnames(final_df_0.05_60) <- c("avg_prob_callback", "avg_intxns_no_vm", "avg_intxns_vm")

no_vm_list_0.1_60 <- c()
vm_list_0.1_60 <- c()

for(i in 1:length(vector_probs)){
  no_vm_list_0.1_60[i] <- call_process(avg_p_callback.no_vm = 0.1,
                                       avg_p_callback.vm = vector_probs[i],
                                       avg_calls_per_hour = 60)[9, 1][[1]]
  vm_list_0.1_60[i] <- call_process(avg_p_callback.no_vm = 0.1,
```

```
                                    avg_p_callback.vm = vector_probs[i],
                                    avg_calls_per_hour = 60)[9, 2][[1]]
}

final_df_0.1_60 <- data.frame(cbind(vector_probs, no_vm_list_0.1_60, vm_list_0.1_60))
colnames(final_df_0.1_60) <- c("avg_prob_callback", "avg_intxns_no_vm", "avg_intxns_vm")

par(mfrow = c(1, 3))

# Lambda = 100

plot(final_df_0.05$avg_prob_callback, final_df_0.05$avg_intxns_no_vm, type="l",
     main="36 sec avg calls",
     xlab="Avg Pr. of Callback | Voicemail",
     ylab="Avg # Interactions/Hour",
     ylim = c(8, 22), lty=2)
lines(final_df_0.05$avg_prob_callback,
      final_df_0.02$avg_intxns_no_vm, type = "l", lty=1)
lines(final_df_0.05$avg_prob_callback,
      final_df_0.1$avg_intxns_no_vm, type = "l", lty=3)
lines(final_df_0.05$avg_prob_callback,
      final_df_0.05$avg_intxns_vm, type = "l", lty=1, col="red")
legend('topleft', c('No VM; P(callback|no VM)=0.02',
                    'No VM; P(callback|no VM)=0.05',
                    'No VM; P(callback|no VM)=0.1',
                    'Voicemail'),
       col = c("black", "black", "black", "red"),
       lty = c(1, 2, 3, 1), bty = 'n', cex = 0.65)

# Lambda = 80

plot(final_df_0.05_80$avg_prob_callback, final_df_0.05_80$avg_intxns_no_vm, type="l",
     main="45 sec avg calls",
     xlab="Avg Pr. of Callback | Voicemail",
     ylab="Avg # Interactions/Hour",
     ylim = c(8, 22), lty=2)
lines(final_df_0.05_80$avg_prob_callback,
      final_df_0.02_80$avg_intxns_no_vm, type = "l", lty=1)
lines(final_df_0.05_80$avg_prob_callback,
      final_df_0.1_80$avg_intxns_no_vm, type = "l", lty=3)
lines(final_df_0.05_80$avg_prob_callback,
      final_df_0.05_80$avg_intxns_vm, type = "l", lty=1, col="red")
legend('topleft', c('No VM; P(callback|no VM)=0.02',
                    'No VM; P(callback|no VM)=0.05',
                    'No VM; P(callback|no VM)=0.1',
                    'Voicemail'),
       col = c("black", "black", "black", "red"),
       lty = c(1, 2, 3, 1), bty = 'n', cex = 0.65)

# Lambda = 60

plot(final_df_0.05_60$avg_prob_callback, final_df_0.05_60$avg_intxns_no_vm, type="l",
     main="60 sec avg calls",
```

```
     xlab="Avg Pr. of Callback | Voicemail",
     ylab="Avg # Interactions/Hour",
     ylim = c(8, 22), lty=2)
lines(final_df_0.05_60$avg_prob_callback,
      final_df_0.02_60$avg_intxns_no_vm, type = "l", lty=1)
lines(final_df_0.05_60$avg_prob_callback,
      final_df_0.1_60$avg_intxns_no_vm, type = "l", lty=3)
lines(final_df_0.05_60$avg_prob_callback,
      final_df_0.05_60$avg_intxns_vm, type = "l", lty=1, col="red")
legend('topleft', c('No VM; P(callback|no VM)=0.02',
                    'No VM; P(callback|no VM)=0.05',
                    'No VM; P(callback|no VM)=0.1',
                    'Voicemail'),
       col = c("black", "black", "black", "red"),
       lty = c(1, 2, 3, 1), bty = 'n', cex = 0.65)
```