Cognitive Modeling (INFOMCM)
General lab information

Structure

You work in pairs on the assignments. Pick someone who has a similar knowledge and skill level to you, so you learn at the same pace. You need to submit your assignment as a PDF file via Blackboard. You do not need to submit code. However, please make sure you have this stored and available in case I have any questions about it.

Please mind these aspects in your report:

- Provide your names and student numbers
- Phrase your answers in clear sentences.
- Make sure that graphs are clear, including the use of appropriate axis labels and a legend.
- If I ask you for a motivation or "why" something is the case, formulate a clear argument. Your argumentation structure is taken into account in grading.

Throughout this assignment, four symbols will be used.

Self-check and hints: At a self-check, you will be asked a question to check whether you understand what you read and did so far. You can discuss this question with your partner. You do not need to report the answers to these questions in your final report. However, of course it can be helpful to note down the answer in a document of your own.
For hints, you will get some more information that might help you for example in answering a question.

Testing: Here we describe concrete tests that you can run to test if your intuition is correct.

Question: This is a question that you need to answer in the file that you hand in through Blackboard.

Engineering: This is an essential step that you need to do in the assignment, and which involves some coding. For example, to make a modification to the model. These steps are typically not directly tied to a question, but are required to be able to answer later questions.

Grading

For each question you can earn points. If you answer all regular questions correctly, you can get at most 10 points (grade: 10).
If you also answer bonus questions, you can earn bonus points. The maximum grade you can get is a 11. There are restrictions on what a high grade can compensate for in the course. See the course manual for more details.

Time management

The assignment has 6 questions. Questions 4, 5, and 6 provide most points and also require the most work. So please plan accordingly. For the average student, this is a suggested time-line. If you are faster than this, we encourage you to continue and also try the bonus assignment:

- Lab 1: Questions 1, 2 and start of 3 (ideally finish after lab)
- Lab 2: Question 4 and 5
- Lab 3: Question 6 (and potentially bonus exercises)

If you are new to R, then please make use of the TAs and lecturers and check with them whether you are "on schedule". We are here to help.
There are ways to do subsets of the questions slightly faster. Please note that I typically provide most points for insight, not for code. So, make sure you gain the insight ☺

Prerequisite knowledge

At the start of this assignment, we make these assumptions about your background:

1. You had some introduction to cognitive modeling (i.e., you attended the lectures, including the general introduction to modeling). If you do not meet this requirement, please check the video lecture and slide material (on introduction to modeling) that is available through Blackboard.

2. You already understand the basics of programming in R. If you have never worked with R before, you should first work through the general introduction to R that is available on Blackboard.

Lab Assignment 1: Processing models

Lecturers

Krista Overvliet (lab coordinator)
Chris Janssen (shadow co-coordinator & creator of assignment)
Teaching Assistants: see course manual
Assistants in data collection (2016): Remo van der Heiden and Ken Beckers

Background

In your later career you might want to model (part of) a cognitive process. For example, to further scientific theory, or to implement a user model in an application. In many cases, you typically start with some preceding theory or an initial model implementation that you then refine. To refine a model for your own purposes, you first need to understand
(1) how this model is structure
(2) how you can compare model results with human data
(3) how you can modify components in the model.

When you then make modifications to the model, you need to understand whether this modification was worth the effort:
(4) did you gain significantly more knowledge about the theory,
(5) and/or did it allow you to apply this model to significantly better situations?
(6)and how does this improvement relate to the amount of modifications that you needed to make (i.e., how many free parameters did you allow yourself in relationship to the improvement in fit).

In this lab session you will gain experience in these aspects of modeling. You will study a model that has been published in the literature. The model (Janssen & Brumby, 2010) describes a situation in which a user switches between two tasks: *steering* a simulated car and *dialing* a pre-rehearsed phone number. The phone number task is a task that has a hierarchical structure and therefore allows for "natural breakpoints" that cue users to interleave (see also Salvucci, 2005; Janssen, Brumby, & Garnett 2012). It is therefore representative for tasks that have a hierarchical structure, such as many office tasks (e.g., see Iqbal & Bailey, 2010)

One critical aspect about the Janssen & Brumby (2010) model compared to other models (e.g., Salvucci, 2005) is that the model allows the modeler to explore what the impact is of various *strategies* for interleaving between two tasks on performance. That is, it does not make a priori assumptions regarding a strategy for interleaving the task. Rather, it allows exploration of what the best strategy might be.

The model was developed for a specific setting with a relatively "old" (non-smart) mobile phone and a simple driving simulator. How well does this model work in a situation with a different phone and a different simulator? We will test this generalizability here.

The model focuses on cognitive processes, with actions defined in the order of steps of 50 to 500 milliseconds. Stated differently, it is at the cognitive band with some rational aspects (in the framework of Newell, 1990). It also focuses mostly on algorithmic questions ("what do people do"), with some implications for the computational level ("why might people do that") (Marr, 1983).

Goals of this lab session

At the end of this lab session, you can:

1. Implement components of a processing model (at the cognitive band / algorithmic level).
2. Implement algorithms to compare model predictions with human data.
3. Evaluate the goodness-of-fit of a cognitive model given (A) the assumptions behind the model, (B) the free parameters of the model, and (C) the quality of the human data.

The level to achieve is that you understand basic concepts, ways of implementation, and evaluation such that you can apply this to your own projects (e.g., your Master's thesis) in more detail for a domain that is of your specific interest. Although you are trained in the R environment, the general principles transcend the programming environment.

General information about the model

In order to get a general idea about the model and the experiment, read Janssen & Brumby (2010), or Brumby, Janssen, Kujala, and Salvucci (2018 – available from Chris upon request). Most relevant information will be in the methods and results section of the model and of the experiment. You can also watch a brief explanation here: https://youtu.be/rQhj0vjVZFU

In addition, below you find some additional figures which are taken from Chapter 3 in Janssen (2012), together with their captions.

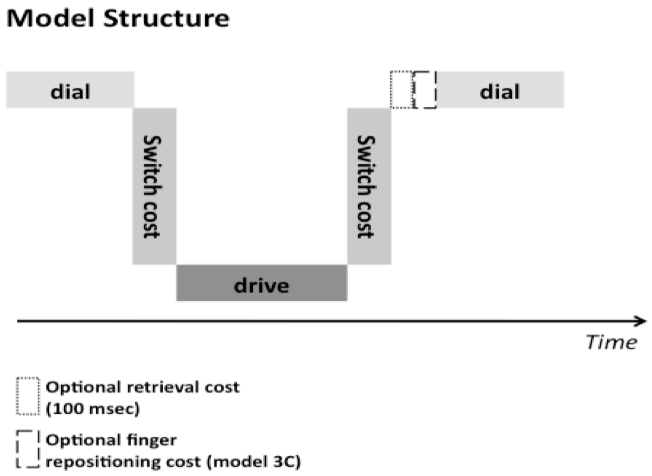


Figure 1: Steps that the model iterates through: dialing, switching attention, driving, switching, and then a series of optional costs (these are only incurred in the Janssen et al 2012 model, see bonus assignments), before continuing to dial. (from Janssen, 2012)

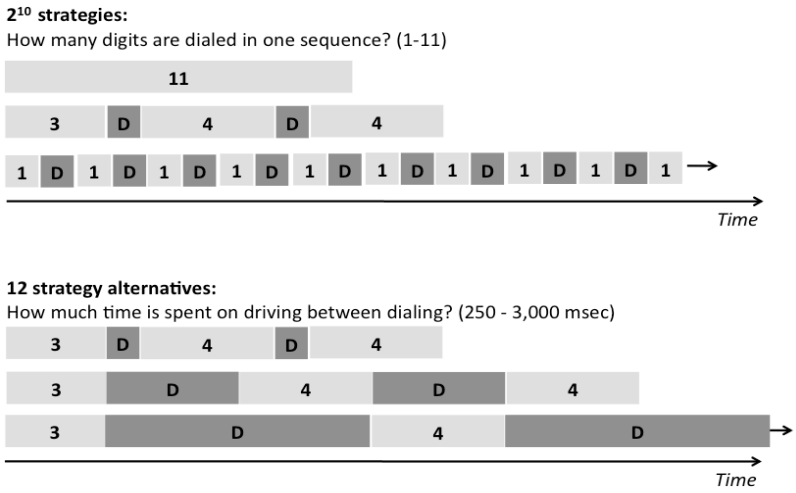


Figure 2: Example of the manipulation of strategy (top: how many digits are dialed before driving?) and strategy alternatives (bottom: how much time is spent on driving?). (from Janssen, 2012)

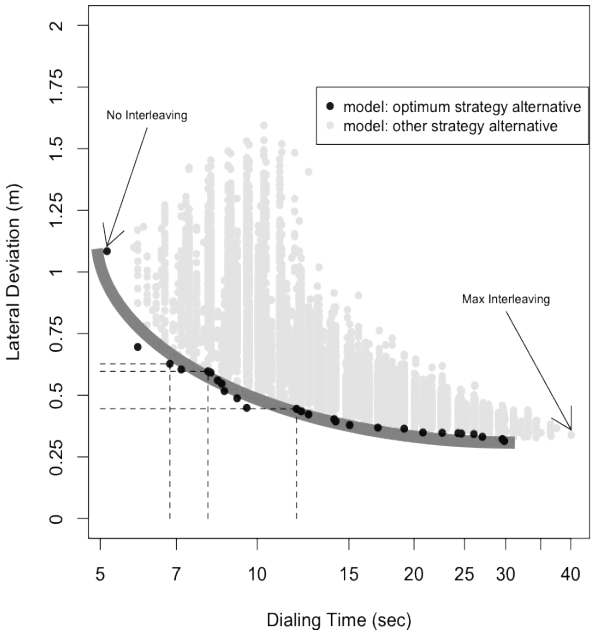
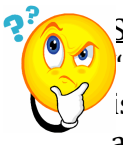


Figure 3: An example of a model performance space. Grey dots are model predictions for strategy alternatives expressed in average lateral deviation (vertical) and total dialing time (horizontal). The black dots are those that are on the pareto frontier. An optimal trade-off curve (black line) can be fitted through these points. (from Janssen, 2012)

Section I: Human Data

We start with inspecting human data. A good understanding of human behavior is essential to ensure that our model is correct and to be able to evaluate its performance later.



Self-check and hints: The relevant human data is stored in the file "keyPressDataWithLaneDeviation.Rdata", which you can load in R. In addition, there is a .csv file if you want to explore the data in a spreadsheet program. In appendix 2, a short description is given of the various columns in this data file. Hints are also given after question 1 below, about specific columns that are useful to look at.

Question 1 (1 Points)

Report the following aspects of the human performance data *for trials in which there are no typing errors made*. Note: below this question I provide hints for each of the subquestions.

A. For the two experimental conditions (steering focus or "dualSteerFocus"; dialing focus or "dualDialFocus") report the mean (*M*) of the total dialing time, standard deviation (*SD*), and standard error of the mean (*SE*) of total dialing time. To do this, make appropriate use of *subsetting*, then calculate the average for each participant per condition (use "aggregate"). Based on these averages you can then calculate the grand mean, SD and SE. In your report you only need to report a table as follows:

	M	*SD*	*SE*
Steering focus	15.0 s	1.5 s	0.4 s
Dialing focus	6.0 s	1.7 s	0.5 s

B. Do the same for average absolute lateral deviation (in m), when you only look at the data points where a key or "." sign was pressed (that is, the start of dialing and the final "." that ends the dialing task; in Janssen & Brumby 2010 a "#" was dialed instead of a ".").

C. Make a plot of how lane deviation changes over time with each keypress, similar to the plot in Figure 1 of Janssen & Brumby (2010). Mind the axis labels, the use of error bars, the use of different symbols for the different conditions, and the inclusion of a legend. You also need to include a figure caption. You do *not* need to indicate in the Figure with text where the chunk boundary is, or where the lane boundary is.

D. Based on the plot in question C, answer the following question: "For each condition argue whether you think the *average* participant waited until the natural breakpoint (between 5th and 6th digit) with interleaving dialing for driving." Motivate your opinion clearly based on the observed pattern.

Hints for question 1:

- General: define your code in a separate function, or in a separate R-file so you have it all in one place.
- General: We only want to consider trials in which NO typing errors were made. You can recognize these trials, based on the value of the column "typingErrorMadeOnTrial". You can look at the *subset* of lines where no correction was made.
- Question A: The combination of the functions "aggregate" and "with" can be useful, as you then later do not need to specify multiple times which exact subset of the table you are looking at.
- Question A: The standardized error of the mean is calculated by first calculating the Standard Deviation and then dividing this value by the square root of the number of participants.
- Question A: to calculate the "total dialing time" make use of these rows:
    - timeRelativeToTrialStart : this indicates the time relative to the start of the trial (i.e., within a trial, this column keeps increasing)
    - phoneNrLengthAfterKeyPress : assume that the typing is done when digit 11 is typed (*not* when the "." is typed and the event "correct" is logged)
- Question B: you need to calculate the absolute (i.e., unsigned) lane deviation. The (non-absolute, signed) lane / lateral deviation is stored in the column "lanePosition". R has a pre-defined function for calculating the absolute value.
- Question B: don't just look at the events "start" and "end", but also when a key is pressed.
- Question B: you need to calculate 1 value per focus condition (that is, average over ALL keypresses, do NOT calculate a value per keypress)
- Question B: the set of datapoints that need to be processed include ALL keypresses, as well as the start and end of trial conditions.
- Question C: make use of R's "aggregate" function to calculate averages first per participant per digit typed and condition. Once you have those averages, then average *across* participants.
- Question C: for lane deviation, you want to take the absolute value of lane deviation.
- Question C: You can use the column "phoneNrLengthAfterKeyPress" to know how absolute lane deviation develops over time.
- Question C: for nice plotting, you might want to look at the help files of "plot", "points", "lines", "par", and "legend"

Section II: Fitting lateral drift of the car

On pages 1555-1556 of the Janssen & Brumby (2010) article, the driving model is described. One sentence is: "*To simulate the driving environment, we permute the vehicle's heading every 50 ms with a value drawn from a Gaussian distribution with M = 0.00 m/s and SD = 0.13 m/s.*"
What is meant here is that every 50 milliseconds the position of the car is changed based on a value that is drawn/sampled from a Gaussian distribution with the provided parameters.

Testing: In R, you can draw random values from a Gaussian function using the function "rnorm". If you have trouble with interpreting what "drawing a value" means, try the function rnorm with the provided *M* and *SD* values. The parameter "n" in this function says how many values are drawn. Look at the written output when you choose n = 10. Do the same for higher n's (e.g., 50, 100, 1000, 10000). When n is really high, it is hard to interpret the values. It might help to make a histogram of the data. For example: first assign the outcome of rnorm to a variable (e.g., *a <- rnorm(10000, 0.0, 0.13)*).
Then plot: *hist(a, col="grey")*
What is the shape of this histogram?
You can also look at a summary of the points by using *summary(a),* or by using other functions such as *mean*, *median*, and *sd*.
What is the mean? What is the median? What is the SD?
How do these values vary with different *n*? (compare especially large and small *n*)

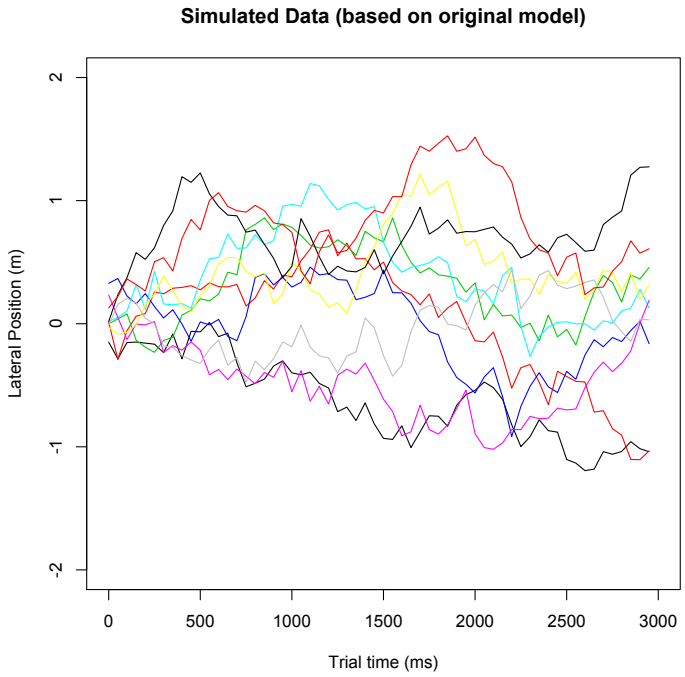
In the experiment of which we use the data during this lab session, we used a different simulator compared to Janssen & Brumby (2010). We can therefore not be sure that the drift of this new simulated environment is similar to that in the original experiment. From now on, you will be calibrating the model to better fit this altered situation.

Your lab instructors collected additional data on 20 trials in which they first actively kept the car in the lane center for approximately 15 seconds, followed by 15 seconds of drift. This data is stored in two files tableOfDriftValuesCalibration.Rdata can be used in R, tableOfDriftValuesCalibration.csv might be useful in a spreadsheet. The columns of these files are explained in appendix 3
(caveat: we use some "dirty hacks" to calibrate the model in the assignment that follows – actual calibration would require even more detailed measurement and analysis that is beyond the scope of this course).

Question 2 (1 Points)
(again, below this question are some additional hints)

A. In the eventual experiment, most drivers typically avoid looking at the road for at most 3 to 4 seconds. Drift during such short intervals can be compared to drifting as measured in the file tableOfDriftValuesCalibration between the time intervals of 15 and 18 seconds.
   Create a plot in which you show how the *lateral position* (posX) of the car changes over time (*trialTime*) for the subset of data between 15000 and 18000 milliseconds (i.e., between 15 and 18 seconds). Make sure that the data for each trial has a unique color.
B. In the paper by Janssen & Brumby (2010) the difference between two consecutive functions has a *M* of 0 and a *SD* of 0.13 meter / second, and is sampled every 50 milliseconds. We can look at what a simulated car's drift

would be if we were to sample from this distribution.
Generate simulated lateral position data for (at least) 20 simulated trials in which you assume that the car starts drifting from a point 0, and samples values from the modeling distribution (M = 0, SD = 0.13) every 50 milliseconds for a period of 3000 milliseconds. Use again different colors for different simulated trials.
For your reference: This is what the plot looks like for me (note that the exact shape of your plot will be different because (i) rnorm uses a random function, and (ii) I only plotted 10 trials, you need to plot more trials):



C. You will probably notice that the shape of the simulated lines is very different from that of the human data. To see this even more clearly, I want you to create two histograms for your report:
(i) one histogram that shows the distribution of car positions as measured in the human trial (i.e., a histogram of the data from Question 2A)
(ii) one histogram that plots the simulated data (i.e., the data from question 2B).

D. For the two histograms you plotted in question C you can also calculate the standard deviation of each of the two datasets. Write the two SDs in your report (clearly indicate about which distribution you are talking: empirical human data or simulated model data).

E. You might notice that the distribution and standard deviation in the resulting data of the model and the human data (i.e., your answers to question 2D) are quite different from each other. To allow a fair comparison between human

and model data, you need to calibrate the model so the parameters for driving are more comparable to those experienced in the simulator. Specifically, you need to change values of the generating model (question 2B, the value SD = 0.13) to get output that looks more like the human data. Try different values in this model (instead of SD = 0.13), and decide which value results in a distribution that is most close to the human data.
In your report include three things:
(i) The final SD to which you want to set the model (what is the value needed to generate this distribution?)
(ii) A plot of how lane position changes over time for the individual simulated trials
(iii) A plot of the resulting distribution
(iv) the SD of this resulting distribution.

Hints for question 2:

- Question A: note that the time in the dataset is given in milliseconds (so 1000 equals 1 second)
- Question B: use the function rnorm to plot the data
- Question B: use a for-loop to run code for multiple trials (e.g., a sequence of trials)
- Question B: The exact position of the car over time is found by adding values over time. You can use R's function "cumsum" to calculate this cumulative sum.
- Question C: Within the function hist() give the bars a color using the "col" argument so it becomes better visible. Use the "breaks" argument (in combination with the seq() argument) to make sure both histograms span the same range and have breaks at the same points. You can also change the range of the y-axis using ylim.
- Question C bullet point (ii) Make sure you store the data that is used to plot each line in question B. Then combine this data into one vector and plot the distribution of values in that vector using a histogram.
- Question E: The stability of the final distribution varies when you change the number of trials that you simulate. The more simulations you run, the more stable your estimate will be. Therefore, feel free to run 50 instead of 20 trials to get an estimate.

Engineering:
You can now also change the value of the drift in the model file. The model file is on Blackboard and called "drivingModel.R". Look in your code where the parameter *gaussDeviateSD* is set, and set it to the value that you calculated in question 2E. Make sure to also keep the old value somewhere (e.g., comment it out), so you can later (in question 6) compare performance of various models.

Important: The driving model makes use of TWO distributions from which driving samples are taken. One distribution focuses on drift while drivers are NOT controlling the car (*gaussDeviateSD*). The other focuses on situations where the driver is ACTIVELY controlling the car (*gaussDriveNoiseSD*). Of course, the assumption is that the car drifts *less* while it is being controlled actively. Therefore, if the value that you set for *gaussDeviateSD* is smaller than the original value for *gaussDriveNoiseSD*, you should also adjust the value of *gaussDriveNoiseSD* such that it is smaller than the value for gaussDeviateSD. Make sure you report both values!

Section III: Fitting Interkeypress intervals (IKIs)

Another difference between the experiment by Janssen & Brumby (2010) and our experiment, is that in our experiment we used a smart phone, that does not have physical keys (only virtual keys). Therefore, the time it takes to type in a key might differ.

In the model, the interkeypressintervals are set for each digit in the parameter "singleTaskKeyPressTimes". As you can see in the model, the assumption is that each keypress takes about the same time. As you have read in Janssen & Brumby (2010), at chunk boundaries of the phone number, the assumption is that some extra time is needed for the retrieval of a chunk. This time is *not* included in the interkeypress interval (in fact, when calculating the time, it was normalized for this time).

Engineering: You need to adjust the parameter for the interkeypress intervals in the model so that it better corresponds to the human data.
In order to do this, calculate for each participant what the average interkeypress interval is between two digits in the single-task trial data that was collected at the end of the experiment (partOfExperiment == “singleDialing2”). Then calculate what on average (across participants), the average interkeypress interval is for the average participant. Based on the reported values, set the interkeypress values in the model to a single-value (i.e., one value that is used for all digits).
Note:
- make sure to also keep the values of the original model, as in a later assignment you will compare performance of a model in which you adjusted the parameter values and one in which you did not.
- you are free to apply some rounding (e.g., round 493 msec to 495 or 500 msec)

Question 3 (0.5 points)
Write down in your assignment the following information:
A. What was the average value of the interkeypress intervals?
B. What value did you pick for your model and why?

Hints for question 3:

- Question A: only calculate interkeypress intervals for the subset of trials that contained NO typing errors.

Section IV: Expanding the strategy analysis

Testing: In the model, a function called "runOneTrial" is defined. This function completed 1 model simulation for 1 specific strategy. Load the entire R code and run the function in the command window, for example with these two commands:

runOneTrial(c(), 5, c(1,6), 11, "07854325698")

runOneTrial(c(4,5), 5, c(1,6), 11, "07854325698")

The first simulation does not interleave dialing for driving at all (i.e., first parameter is the empty vector c()). The second simulation does some driving before the 4th and the 5th digit.

Each time when the model switches, it makes 5 steering updates of 250 msec each. It does this for a phone number where the chunk boundaries are before the 1st and 6th digit, that has 11 digits in total, and of which the phone number is "07854325698".

Make sure that you understand the output of the model. I have provided some questions below to guide you in understanding. When you answer these for yourself, it might help to go through the code and see how time is added for example to the list "times". It might also help to compare the output with the pictures of the model structure that were provided at the start of this assignment.

Particularly, you can look at the following aspects in your model output and compare it with the model code:
- what is plotted over time?
- how much time is there between two keypresses?
- why is the total trial time the amount that it is, given the parameters for switch costs and interkeypress intervals (NB: it might help to first explore this for a model that does no interleaving and then for various models that interleave at various locations and with varying number of steering updates)

Self-check and hints: The model includes some parameters that are variables. Specifically, the position of the car depends on values that are drawn from Gaussian distributions (see for example code in the functions *calculateLaneDrift* and in *velocityCheck*). In order to get a stable estimate of what performance might look like, we want to run multiple simulations of the model.

The function "*runAllSimpleStrategies*" runs multiple simulations (number set with parameter "*nrSimulations*") for a specific phone number. In addition, it also tests various strategies, namely strategies that interleave before every digit, before every 2nd digit, before every 3rd digit, before every 4th digit, etc.

Make sure you understand what the function *runAllSimpleStrategies* does. For example, test what the output looks like when you run 1 simulation and when you test 5 simulations.

Can you guess how many data points are plotted in each graph? Why is this a specific number? (hint: relate this to strategies for interleaving and strategy alternatives based on steering updates)

Question 4 (1.5 points)

Run the function *runAllSimpleStrategies* multiple times: once with 1 simulation, once with 5 simulations, once with 10 simulations, and once with 50 simulations. If your computer can process it (depending on processing speed), also run a model with 100 or 200 simulations. Note: the goal of this lab is for you to get insight in modeling, NOT to have your computer run simulations for hours. Check the hints below this questions for ways to keep the computation time within bounds.

A. Plot the graphs of the various simulations in your report (i.e., 4 graphs if you did 1, 5, 10, and 50 simulations). Clearly label each graph to show how many simulations you used.
B. Explain the following: what would be a good number of simulations for your model and why? Relate your answer to your findings in question A and take into account that running 1000s of simulations is not an option; you have to make some choices.

Hints for question 4:

- General: as of now you will start some simulations that will demand computing power, and therefore time, of your machine. At the time of this writing, the default R installations do not use multi-core computing. The "Microsoft Open R" version of R DOES allow multi-core computing, and can therefore be faster in its calculations. A downside of this version is that not all R packages are compatible. You can download Microsoft Open R here: https://mran.microsoft.com/open/
- General: in Chris' MacBook Air (1.7 GHz processor, 8GB RAM) with regular R (not open R) it takes about 30 seconds to run 50 simulations, 1:20 minutes to run 100 simulations, and 4:30 minutes to run 200 simulations. If your computer is a lot slower than Chris' computer, running 200 simulations should not be your goal! (unless you're up for a coffee break ☺)
- General: if you consider all strategy alternatives (i.e., drive values of 250, 500, 750, … 3000 milliseconds), it will take a while to run the model. To get results faster, you can only look at an interesting subset of alternatives (for example, 250, 500, 1000, 1500, 2000, and 3000 msec).
    - You can set the subset using this part of the R-code: *steeringTimeOptions <- c(1,2,3,4,5,6,7,8,9,10,11,12)*
    - "1" here means "drive for 1 x 250 milliseconds; "12" means "drive for 12 x 250 msec = 3000 msec"
    - For example, change this to: steeringTimeOptions <- c(1,2,4,6,8,12)
    - Document in your assignment what you choose (So I know why there are some deviations from an "expected" results plot)

- General: To further speed up your code: check whether you are *not* using the command "rbind" to bind each new row to a general table. Instead make some local tables first and then add the local table to the "global" table once you are done.

Engineering: The model only explores a limited set of strategies for interleaving dialing and steering. Create a new function called "*runAllComplexStrategies*" which is largely based on the function "*runAllSimpleStrategies*". However, the new function should be able to explore ALL possible strategies of interleaving digits.
That is, instead of only looking at regular/consistent interleaving patterns such as:
0-7-8-5-4-3-2-5-6-9-8
07-85-43-25-69-8
0785-4325-698
078-543-256-98
(in which the model always types a consistent set of digits before interleaving for driving)

Your model should run all strategies that look at all unique ways of interleaving where the length of the set of digits before interleaving can vary, for example:
0785-4325698
07-854325-698
0785432569-8

Stated differently, write a piece of code that systematically goes through all positions (e.g., using a for-loop for example) and for those positions, it should systematically interleave (or not) at those points on different runs.

Make sure that you list the strategy in a specific column (or columns) in your model output, so you can later find all strategies that interleaved at specific positions, such as only at the natural breakpoint. The model should also vary how much time is spent on steering (similar to the previous function). This should therefore also be part of your model output.

By the way: it might be valuable to only test your model with 1 or 2 simulations at a time instead of 50, to save processor time while you are creating (and debugging) your code.

The model output should again produce a graph of the performance space, and you should be able to store the model data in a data frame, so you can analyze it later in more detail.

Self-check and hints:
- If you were only able to plot strategies that interleave solely at the "natural breakpoint" (in between the 5th and 6th digit), how many data points would you plot?
- How many strategy alternatives are plotted in this graph and why?

Question 5 (3 points)

Run your revised model with at least 50 simulations per strategy alternative (or more, if you have a powerful computer, or less if you find out that running the simulations takes a lot of time --> let your lecturer know if this is the case, your code might not be efficient).

Then answer these questions:

A. Plot a Figure like Figure 4 in Janssen & Brumby (2010). Make sure to include the axis labels, make sure to highlight the strategies that ONLY interleave between the fifth and sixth digit (i.e., only at the natural breakpoint, similar to the paper). Make sure to also include the human means with standardized errors of the mean around the points (use your data from question 1).

B. In Janssen & Brumby (2010) and Janssen, Brumby, & Garnett (2012), it was argued that participants' performance fell on the outside edge of the "performance space". This was used as an argument to consider participants' performance to be efficient. Argue, using your graph, whether that is also the case in the new data.

C. Another argument in Janssen & Brumby (2010) is that in the steering focus condition people do not always interleave solely at the natural breakpoint (i.e., not solely in between digits 5 and 6). You can explore this with your model. Look which model strategies result in performance that falls inside square area defined by the error bars of the human data. Specifically, you can use the data to check how frequently these model strategies interleave, and at which position they do the interleaving for the very first time (i.e., before or after the natural breakpoint?). Conduct such an analysis and argue whether it is plausible (or not) that in the steering focus condition our participants interleaved *solely* at the natural breakpoint of the phone number.

Hints for question 5:

- General: All the considerations that were given as hints with question 5 also apply here.
- General: To further speed up your code, consider running fewer simulations. However, make sure it is at least 10, and preferably at least 25 to 30 per strategy alternative.
- General: Make sure you can access your code / data after the simulation is done (don't just plot it and save the plot; save the data as well). You can do this in various ways. One way is to add a line of code that saves the data within your function. However: make sure that this does not override any existing files / data.
    - Another way might be to run code within a function (i.e., don't run the function with a function call, but run the code within the function after

you initialized variables). Chris can explain this during lab if it is confusing. (for some this method is intuitive, for others it is confusing)

- General: another way to speed up the model is by not storing all data all the time. Specifically: at some point you run through all simulations in a line like this "*for (i in 1:nrSimulations)*". The code currently stores all the data of each simulation in multiple long vectors (see the details of this code segment for *runAllSimpleStrategies*). This can be improved:
    - In the end we only need the average/mean values over the many simulations.
    - So, one thing you could do is: reset the "local" variables that store the details (e.g., position of the car at each keypress) right before this for-loop.
    - Then store all the details of the for loop, but once one set of simulations is run (i.e., at the end of the "for (i in 1:nrSimulations)") calculate the relevant average metrics.
    - Store these average metrics, not the details.
    - This should save you from storing all data. You lose some detail, but keep the mean (and, for now, most interesting) values.

Section V: Relationship between model flexibility and model fit

In this assignment you have so far made several changes to the model:

1. You have calibrated the simulated car's drift function to the new simulator.
2. You have calibrated average typing time to the observed single-task typing times of an average participant.
3. You have looked at what the impact is of running more or fewer observations.
4. You have explored a wider space of strategies.

The more a model is informed by theory, the stronger its prediction is. The more a model is adjusted/fitted to a specific experiment, the larger the danger that you are "overfitting" the model to this specific situation.

Roberts and Pashler (2000) discuss at great length that there needs to be a fine balance between the flexibility of the model ("number of free parameters") and the goodness of the fit.

Engineering: Explore in your model what the impact is of various parameter changes that you made on the predictions of the model. Specifically, explore systematically what the impact is of:

- Calibrating the drift function to this data set VERSUS leaving the drift function at the old value of Janssen & Brumby (2010) *(2 options)* (NB: check the text on "engineering" after question 2 and report whether you only changed 1 SD value or 2 SD values)
- Calibrating the participants IKI to the single-task values VERSUS leaving them at the default function of Janssen & Brumby (2010) *(2 options)*

- Modeling a low number of simulations (10 per run) VERSUS running a high number of simulations (e.g., 50 per run) *(2 options)*

In total, this gives you 8 unique models (2 x 2 x 2). When possible, run this with the function "*runAllComplexStrategies*". However, if your computer has performance issues running this code, you might want to look only at the function "*runAllSimpleStrategies*", as long as you extend this function such that it includes a strategy that interleaves only at the natural breakpoint in the phone number.

Important: With the model results, you need to do some analyses (see question below). Therefore, make sure that you SAVE the model output to a unique file, so you can later always load it and compare it with other model output.

Question 6 (3 points)
Generate the plots for the 8 model situations above & include those in your report. Then answer the following question: how crucial is each manipulation for (1) your overall model fit, (2) the general conclusion that people do not solely interleave at natural breakpoints in the steering focus condition.
In your answer, you should use (i) your model results, (ii) your knowledge about model fit as described in the paper by Roberts and Pashler, (iii) your knowledge about the experimental set-up that was used and how this compares with the experimental design of Janssen & Brumby (e.g., factors that might have influenced the stability and reliability of the result).

Hints for question 6:

- General: All the considerations about "speeding up" your code from questions 4 and 5 apply.

Bonus assignments (at most 1 point extra)

If you are up for a challenge, you can try one or more of these bonus assignments. The assignments vary in difficulty and therefore also in how many points they provide you with. If you want to work on one of these assignments, please explain clearly what you did in your lab report. However, I do not want to read code or pseudo-code; explain it using your words :-)
Each assignment will have specific a specific question or questions that you need to answer. Please note that we will be relatively strict in marking. Simply "just trying out" something will not give you (partial) credit.

Bonus option 1: Individual differences in typing speed (medium)

In the model above we only looked at average interkeypress intervals for the average participant. However, some participants might have typed faster than others. Make a model that has a *different average* typing speed for *different* individuals. Specifically: calculate for each participant what their individual average typing speed is. Then run a model simulation for that participant using their average typing speed (you can use a constant value for each digit).
In your report, you should include a plot that shows the trade-off space for three individuals (e.g., in a similar fashion to Figure 4 in Janssen & Brumby, 2010):
- the slowest typing individual
- the fastest typing individual
- an individual with an average typing time

Make sure to describe which individual this is, what their typing speed was and what value you set the model to.
Then argue whether calibrating the model for individual differences is valuable, given the balance between having an extra free parameter and potential increase in fit.
(NB: It might be valuable to also plot the performance of these individuals on the general model space of the "average" participant. Perhaps the individual's data points falls outside the performance space of the average model, but inside the performance space of the individually calibrated models).

Bonus option 2: Variable key-press times 1 (medium)

In Janssen, Brumby, & Garnett (2012), the proposal is that it takes longer to type some digits compared to others, and that these digits might also function as "natural breakpoints". You're already moving your hand/finger quite far between some keys and that might encourage you to switch keys.
Adjust the model such that it also has such additional costs for digits that are far apart and see whether there is an effect of interleaving at positions where the keys are far apart ('motor cues') and at cognitive natural breakpoints .
- Report what values you set the various IKIs to
- Report results using a Figure like Figure 4 in Janssen, Brumby, and Garnett (2012).

- Then argue whether calibrating the model for these motor cues is valuable, given the balance between having an extra free parameter and potential increase in fit.

Bonus option 3: Variable key-press times 2 (medium)
In your current model, the model always takes the same amount of time for typing 1 digit. However, in practice people vary from trial-to-trial. Stated differently, our keypress times can be considered as being drawn from a distribution of values with a mean and a standard deviation. Make a model in which on each trial for each interkeypress interval the IKI time is a value that is drawn from a distribution with a mean and SD similar to the one observed in the human data. In your report:
- report what the mean and SD are that you used
- show a model result plot (similar to Figure 4 in Janssen & Brumby, 2010)
- Then argue whether calibrating the model for this variability is valuable, given the balance between having an extra free parameter and potential increase in fit.

Bonus option 4: Different phone structures (easy-medium)
In the experiment that was run, the phone number only had 1 natural breakpoint. What do model results look like for phones that have a very different structure, for example more like an American (and Utrecht) phone number?
XXX - XXX - XXXX
Or a structure where there are frequent breakpoints:
XX- XX -XX - XX - XX

Run models for phone numbers of different structures (make sure they all have a similar total digit length). Then:
- describe in your report what you did
- show model results plots for each model (similar to Figure 4 in Janssen & Brumby, 2010)
- Argue what the impact is of phone structure (specifically natural breakpoints FREQUENCY and POSITION) on what the performance space looks like.

Bonus option 5: Own ideas (easy-difficult)
You might have developed some ideas yourself regarding further testing or refinement of the model. You are free to do those as well. Please discuss them with Chris first, so he can help you manage and make sure that this is something valuable to do.

References

Brumby, D. P., Janssen, C. P., Kujala, T., & Salvucci, D. D. (2018). Computational Models of User Multitasking. In: Oulasvirta, Kristensson, Bi, & Howes (eds.) *Computational Interaction Design*. Oxford University Press. Available from Chris upon request

Iqbal, S. T., & Bailey, B. P. (2010). Oasis: A framework for linking notification delivery to the perceptual structure of goal-directed tasks. *ACM Transactions on Computer-Human Interaction (TOCHI)*, *17*(4), 15:1–28.

Janssen, C. P. (2012). *Understanding strategic adaptation in dual-task situations as cognitively bounded rational behavior*. London, United Kingdom: UCL. PhD thesis

Janssen, C. P., & Brumby, D. P. (2010). Strategic Adaptation to Performance Objectives in a Dual-Task Setting. *Cognitive Science*, *34*(8), 1548–1560. http://doi.org/10.1111/j.1551-6709.2010.01124.x

Janssen, C. P., Brumby, D. P., & Garnett, R. (2012). Natural Break Points The Influence of Priorities and Cognitive and Motor Cues on Dual-Task Interleaving. *Journal of Cognitive Engineering and Decision Making*, *6*(1), 5–29. http://doi.org/10.1177/1555343411432339

Marr, D. (1982). Vision: A Computational Investigation into the Human Representation and Processing of Visual Information. San Francisco, CA: Freeman.

Newell, A. (1990). Unified theories of cognition. Cambridge, MA: Harvard University Press.

Roberts, S., & Pashler, H. (2000). How persuasive is a good fit? A comment on theory testing. *Psychological Review*, *107*(2), 358–367.

Salvucci, D. D. (2005). A Multitasking General Executive for Compound Continuous Tasks. *Cognitive Science*, *29*(3), 457–492. http://doi.org/10.1207/s15516709cog0000_19

Appendix 1: Brief description of the procedure of the experiment

In the experiment we largely followed the procedure in Janssen & Brumby (2010). However, to not make it too long, we did a faster procedure. The participants (students of the course) worked on these aspects in order:

1. Practice the first chunk of the number for 1 minute, by typing it on the phone
2. Practice the second chunk for 1 minute
3. Practice the combination
4. Practice single-task driving
5. Approximately 5 single-task dialing trials (first time) while seated in the simulator. Stored in the "keyPressDataWithLaneDeviation" as partOfExperiment == "singleDialing1"
6. Approximately 5 single-task driving trials of 20 seconds each. NB: this data is logged, but not clearly labelled in the driving data file. If you want to work with these data, please ask Chris.
7. Approximately 10 dual-task dialing-while-steering data with priority A (either steering, or dialing).
8. Approximately 10 dual-task trials with the other priority.
9. Approximately 5 single-task dialing trials (second time) while seated in the simulator. Stored in the "keyPressDataWithLaneDeviation" as partOfExperiment == "singleDialing2"

If you want to know more, please ask Chris.

Appendix 2: Columns in the file "keyPressDataWithLaneDeviation"

There are two versions of the file "keyPressDataWithLaneDeviation".

1. The .Rdata file is faster to load in R. You should try to use this file.
2. The .csv file might be convenient to inspect the data in a spreadsheet program.

R stores data in rows, with consecutive rows being later in time (and for later participants).

There are multiple columns in the data frame. They don't have a logical order, as it is partially constrained by the driving simulator output. Some columns are needed to generate other columns in preprocessing steps (which your lecturer did).

These are the columns:

- time: a date and time, with accuracy up to milliseconds
- pp: the participant ID (a number)
- Event1: column that indicates whether the trials starts (value: "Start", this is when the first "." was pressed), or ends ("Correct": when the last "." was pressed), or whether a key is pressed ("Keypress")
- PhoneNrSoFar: the string of the phone number **before** the key was pressed
- Event2: has value 190 when Event1 is "Start" or "Correct". For Event1 values "Keypress" it indicates *which* key is pressed
- trial: counts the trials of the experiment, starting with 1 for each new participant
- typingErrorMadeOnTrial: has value "1" if a typing error is made on the trials. In most analyses, you might not want to include these rows.
- partOfExperiment: what is the participant working on? "singleDialing1" (first encounter with single-task dialing), "dualSteerFocus" (dual task trials where steering was prioritized), "dualDialFocus" (dual task trials where dialing was prioritized)
- timeConverted: very similar to column "time" except that each ":" has been replaced by a "."
- timeInMsec: the number of milliseconds relative to 1-1-1970 at 00:00.000
- lanePosition: for trials where there was driving, the lane position of the car at that point in time
- phoneNrLengthBeforeKeyPress: how long is the phone number string BEFORE the current key was pressed?
- timeRelativeToTrialStart: how many milliseconds passed since the first "." was pressed (trial start)
- phoneNrLengthAfterKeyPress: how long is the phone number after they key was pressed? You can use this in combination with the column "lanePosition" to plot the laneposition for each keypress

Appendix 3: Column names in the file "tableOfDriftValuesCalibration"

Again, there is a .Rdata and a .csv file.

These are the meanings of the columns:

- time: time in milliseconds since 1-1-1970 at 00:00.000
- posX: the lateral deviation in meters
- posY: the height of the car in meters (as you might have seen, at the start of the trial, the car "drops out of the sky")
- posZ: the distance that is covered during a trial in meters. Note that a trial starts at approximately distance 24400, and that this distance gets smaller during the trial
- rotationX, rotationY, rotationZ, rotationW: other parameters from the simulator (to ignore for now)
- speed: speed of the car in km/h
- steerPosition: angle of the steering wheel
- gasPosition, brakePosition: whether the gas pedal and brake pedal were pressed and how strong (these are not pressed in this trial)
- engineRunning: is the engine on and the trial running?
- driver: who is driving? Driver A or driver B?
- normalizedTime: time in milliseconds where I subtracted some of the larger values, so it becomes readable again
- trial: which trial is running. Trial 1-10 is from driver A, trial 11-20 is driver
- trialTime: the time of the trial since the trial start in milliseconds