Lista para Modelo Dinâmico

Econometria II - Prof. Victor Gomes

Thiago Mendes Rosa

05/07/2019

```
# Informações da sessão do R em que esta lista foi realizada:
# R version 3.5.3 (2019-03-11)
# Platform: x86_64-w64-mingw32/x64 (64-bit)
# Running under: Windows 10 x64 (build 18362)

# Versão do R disponível em:
# https://mran.microsoft.com/download

# Alguns pacotes, para certas plataformas,
# podem solicitar a instalação do Rtools. Ele está disponível em:
# https://cran.r-project.org/bin/windows/Rtools/

# Para rodar este documento com o Rmarkdown foi utilizada a
# versão RStudio 1.2.1335 - Windows 7+ (64-bit) disponível em:
# https://www.rstudio.com/products/rstudio/download/

# Utilizou-se o MiKTeX console com a versão 2.9.6751.
# Para o relatório rodar corretamente, é necessário que
# ele esteja configurado com a opção
# "Always install missing packages on-the-fly"

# Todos os softwares e pacotes utilizados são gratuítos

# Lista de pacotes a utilizados
# Criar um objeto com os pacotes
pacotes <- c("tidyverse","data.table","tabulizer","knitr","ggplot2",
             "pracma","nnet")

# Instalar pacotes, caso não estejam installados
if (length(setdiff(pacotes, rownames(installed.packages()))) > 0) {
```

```
  install.packages(setdiff(pacotes, rownames(installed.packages())))

} else {"Todos os pacotes estão instalados"}
```

```
## [1] "Todos os pacotes estão instalados"
```

```
# Carregar pacotes necessários
library(tidyverse)
library(data.table)
library(tabulizer)
library(knitr)
library(ggplot2)
library(pracma)
library(nnet)

# Ajustar opção do chunk para exibição dos resultados
knitr::opts_chunk$set(echo = TRUE)
```

1 Teoria

1.1 Modelo Econômico

Harold Zurcher gerencia uma frota de ônibus que é sujeita a todo tipo de problema quando esta na rua. A milhagem (quilometragem) acumulada de um ônibus x_t é a variável de estado do problema. O desgaste do ônibus afeta o custo operacional esperado $c(x_t; \theta_1)$ que depende da milhagem e um vetor de parâmetros não conhecido $\theta_1 = \{\theta_{11}, ..., \theta_{1n}\}$.

Assuma que os custos dos ônibus vêm de dois componentes: manutenção regular e despesas operacionais $m(.)$ e o custo $f(.)$ de substituir o motor no caso de falha (que é um evento estocástico que ocorre com alguma probabilidade).

a) Escreva o custo como uma combinação destes dois componentes. Indique claramente quais elementos são função de x_t. Argumente informalmente que $\frac{\partial c}{\partial x_t} > 0$. Esta hipótese é necessária para a solução do modelo. Ela é uma boa hipótese? Você pode fazer um argumento para explicar por que $\frac{\partial c}{\partial x_t}$ poderia ser negativa (pelo menos para alguns valores de x_t)?

Considere que o custo $m(.)$ pode ser decomposto em $m(m_r(x_t), m_o(.))$, em que m_r denota as despesas com manutenção regular e m_o as despesas operacionais. A função custo pode ser escrita como $c = (m(m_r(x_t), m_o(.)), f(x_t, .))$. Espera-se que, quanto maior for a utilização de um ônibus da frota, maiores serão tanto os custos de manutenção regular quanto os custos para substituição do

motor em caso de falha (numa eventual retifica do motor), i.e. $\frac{\partial c}{\partial x_t} > 0$*. Com isso, quanto mais o ônibus roda, maior é o desgaste de suas peças e, portanto, maiores serão os custos. O custo poderia ser não positivo com* x_t *em seus valores iniciais se, por exemplo, houvesse uma garantia do fornecedor para qualquer tipo de problema nas milhas iniciais ou pacotes de manutenção até certo número de milhas inclusos com a compra do veículo.* □

b) Rust não possui dados suficientes para estimar os vários componentes da função custo, então ele simplesmente estima $c(x_t, \theta_1)$**. Que dados ele precisaria e que estratégia ele poderia empregar se fosse querer identificar separadamente** $m()$ **e** $f()$**? Descreva qualquer hipótese que você precisa usar com os dados para fazer funcionar a sua solução.**

Rust afirma que não estavam disponíveis dados detalhados de manutenção e dos custos que envolvem a perda de clientes pela eventual falha de funcionamento nos ônibus, um evento estocástico. Assim, não seria possível decompor o custo como:

$$c(x, \theta_1) = m(x, \theta_{11}) + \mu(x, \theta_{12}) b(x, \theta_{13})$$

em que (x, θ_{11}) *é a esperança condicional das despesas normais com manutenção e operação,* $\mu(x, \theta_{12})$ *é a probabilidade condicional de uma falha inesperada no motor e* $b(x, \theta_{13})$ *é a esperança condicional dos custos relacionados a troca do motor (reboque, troca de peças e perda de confiança do consumidor). Assim, Rust estima apenas a soma desses componentes,* c*. Como a solução é especificar e estimar a soma dos custos, é preciso que estes custos sejam efetivamente separáveis e não sejam dependentes entre si.* □

No começo de cada período t Harold Zurcher observa x_t e decide quando pagar o custo de manutenção $c(x_t, \theta_1)$ ou substituir o motor, que instantaneamente leva a medida de uso para zero $x_t = 0$. Então, o benefício de um único período é dado por

$$u(x_t, i_t, \theta) = \begin{cases} -c(x_t, \theta) & \text{if } i_t = 0 \\ R - c(0, \theta) & \text{if } i_t = 1 \end{cases}$$

tal que R é o custo esperado de substituição do motor e i_t é a decisão binária de investimento.

Faça $F(x_{t+1}|x_t, i_t, \theta)$ representar a função de distribuição cumulativa do processo estocástico que governa a evolução da variável de estado.

c) Escreva a equação de Bellman para este problema.

Considere que o processo estocástico governando $\{i_t, x_t\}$ *é solução para o seguinte problema de parada ótima regenerativo:*

$$V_\theta(x_t) = \sup_{\Pi} \mathbb{E}\left\{ \sum_{j=t}^{\infty} \beta^{j-t} u(x_j, f_j, \theta_1) | x_t \right\}$$

com a função utilidade definida anteriormente, Π *sendo a sequência infinita das regras de decisão* $\Pi = \{f_t, f_{t+1}, ...\}$*,* f_t *sendo a decisão de troca do motor no período* t*, função de toda a história do processo* $i_t = f_t(x_t, i_{t-1}, x_{t-1}, i_{t-2}, x_{t-2}, ...)$*. A expectativa anterior é tomada com respeito ao processo estocástico controlado* $\{x_t\}$*, cuja probabilidade é denotada por* F *(conforme definição anterior), cuja probabilidade de transição é:*

$$F(x_{t+1}|x_t, i_t, \theta) = \begin{cases} \theta_2 exp\{\theta_2(x_{t+1} - x_t)\} & \text{if } i_t = 0 \quad \text{and } x_{t+1} \geq x_t \\ \theta_2 exp\{\theta_2(x_{t+1})\} & \text{if } i_t = 1 \quad \text{and } x_{t+1} \geq 0 \\ 0 & \text{otherwise} \end{cases}$$

A equação acima tem a seguinte lógica: a milhagem acumulada advém de uma distribuição exponencial quando o motor é mantido, considerando a diferença entre o período futuro e o atual; por outro lado, quando a troca é efetuada, a milhagem se regenera e a distribuição exponencial da milhagem acumulada é função apenas do período futuro.

A função definida anteriormente, $V_\theta(x_t)$*, é o valor da função, sendo solução única para a equação de Bellman, dada por:*

$$V_\theta(x_t) = \max_{i_t \in C(x_t)} [u(x_t, i_t, \theta_1) + \beta \mathbb{E} V_\theta(x_t, i_t)]$$

em que $C(x_t) = \{0, 1\}$ *e* $\mathbb{E}V_\theta(x_t, i_t)$ *é definido como:*

$$\mathbb{E}V_\theta(x_t, i_t) = \int_0^\infty V_\theta(y) p(dy|x_t, i_t, \theta_2) \quad \square$$

Se pode obter uma solução analítica para a equação de Bellman em **c)** assumindo que a distância percorrida em cada período é distribuída exponencialmente com o parâmetro θ_2, que é independente da milhagem rodada no período anterior:

$$F(x_{t+1} - x_t) = 1 - exp(-\theta_2(x_{t+1} - x_t))$$

Fazendo isso implica em uma função política estacionária

$$i(x_t, \theta) = \begin{cases} 1 & \text{if } x_t \geq \gamma(\theta) \\ 0 & \text{if } x_t < \gamma(\theta) \end{cases}$$

tal que $\gamma(\theta)$ é a solução única para

$$R(1-\beta) = \int_0^{\gamma(\theta)} [1 - \beta exp(-\theta_2(1-\beta)y)] \frac{\partial c(y,\theta_1)}{\partial y} dy$$

Caso esteja interessado em detalhes deste modelo, ele foi derivado por Rust em um paper anterior ("Stationary Equilibrium in a Market for Durable Assets", Econometrica 1985).

d) Este modelo sofre de um problema de "sobre-previsão". Explique o que significa. Se pode usar este modelo para estimação? Por que ou por que não?

Esta especificação implica em uma função de risco degenerada, na qual, após certo limiar, sempre é realizada a troca do motor ($x_t \geq \gamma(\theta)$). Todavia, o autor aponta que os dados refutam tal hipótese, uma vez que a troca do motor ocorre numa faixa de milhagem muito ampla (de 82.400 a 387.000), sendo incompatível com uma regra de limiar a partir do qual a troca sempre ocorre. Nesse sentido, ao utilizar tal modelo, teríamos um ponto a partir do qual o motor do ônibus é sempre trocado, algo não aderente a realidade. Assim sendo, a utilização desse modelo para previsão não parece ser o mais adequado. Como o autor aponta, a milhagem dos ônibus (x_t) não parece ser apenas o único determinante para a troca do motor, existindo um conjunto de outros fatores que podem influenciar tal decisão (o que levará a adição do termo ε_t ao modelo). Além disso, assume-se que $(x_{t+1} - x_t)$ tem uma distribuição exponencial i.i.d., fato refutado pelo conjunto de dados analisado. □

1.2 Modelo Estocástico

Para estimar o comportamento que vemos nos dados é preciso adicionar termos de erro ao modelo. Uma forma de fazer isso é assumindo que os agentes desviam da solução ótima do modelo:

$$i_t = i(x_t, \theta) + \omega_t$$

Rust opta por uma estratégia diferente. Ele assume que os payoffs recebem choques aleatórios $\varepsilon_{it,t}$:

$$U(x,i,\theta) = \begin{cases} -c(x_t,\theta_1) + \varepsilon_{0,t} & \text{if } i_t = 0 \\ R - c(0,\theta_1) + \varepsilon_{1,t} & \text{if } i_t = 1 \end{cases}$$

$\varepsilon_{it,t}$ é observado por Harold Zurcher, mas não pelo econometrista.

e) Interprete ω_t e ε_t. Que tipo de erro eles representam? Discuta as vantagens e desvantagens de cada escolha de modelagem. Por que você acha que Rust optou pela segunda opção?

Em ω_t, tem-se um componente com variáveis de estado que influenciam a decisão tomada pelo agente, mas desconhecida pelo econometrista. O caso de ε_t é análogo, mas ele entra na função

utilidade, como sendo um componente da alternativa i no período t que, igualmente, é conhecido pelo agente mas desconhecido pelo econometrista. Rust argumenta que a primeira opção seria internamente inconsistente, uma vez que o modelo estrutural partiu da hipótese de que o comportamento do agente é compatível com a solução de um problema de otimização dinâmico. A segunda abordagem trata o termo estocástico de modo que ele seja internamente consistente, sendo incorporado explicitamente na solução do problema, com a interpretação de que ε_t é uma variável de estado não observável pelo econometrista, mas pelo observada pelo agente. O autor comenta ainda que essa estratégia garante um benefício adicional ao abrir a possibilidade de que mais parâmetros possam ser estimados. □

f) Escreva a nova função-valor $V(x_t, \varepsilon_t)$

$$V_\theta(x_t, \varepsilon_t) = \sup_{\Pi} \mathbb{E}\left\{ \sum_{j=t}^{\infty} \beta^{(j-t)}[u(x_j, f_j, \theta_1) + \varepsilon_j(f_j)] | x_t, \varepsilon_t, \theta_2, \theta_3 \right\}$$

Em que $\Pi = \{f_t, f_{t+1}, f_{t+2}, ...\}$, $f \in C(x_t), \forall t$, que é o conjunto de escolha, e a esperança é tomada com respeito ao processo estocástico controlado $\{x_t, \varepsilon_t\}$ cuja densidade de probabilidade é definida a partir de Π e a transição de probabilidade p é dada por:

$$dp\{x_{t+1}, \varepsilon_{t+1}, ..., x_{N+1}, \varepsilon_{N+1} | x_t, \varepsilon_t\} = \prod_{i=t}^{N-1} p(x_{t+1}, \varepsilon_{t+1} | x_i, \varepsilon_i, \theta_2, \theta_3) \quad \square$$

A solução para este problema é dada por uma regra de decisão estacionária $i_t = i(x_t, \varepsilon_t; \theta)$, que especifica a decisão ótima do agente quando a variável de estado é (x_t, ε_t). Neste ponto, o modelo estatístico é completamente especificado mas ainda é muito complicado de estimar – especialmente com dados limitados.

1.3 Simplificando as hipóteses para estimação

Rust fez as hipóteses simplificadoras $P(x_{t+1}; \varepsilon_{t+1} | x_t, \varepsilon_t) = P_1(x_{t+1} | x_t) P_2(\varepsilon_{t+1})$ (ele chama isso de “Conditional Independence Assumption”).

g) Defina o valor esperado condicional de $V(x_t)$ sobre ε como

$$EV(x_t) = \int V(x_t; \varepsilon_t) P_2(\varepsilon_t)$$

Mostre que

$$EV(x_t) = \mathbb{E}_t[\max\{-c(x_t, \theta_1) + \varepsilon_{0,t} + \beta \int EV(x_{t+1}) P(dx_{t+1} | x_t);$$
$$-R - c(0, \theta) + \varepsilon_{1,t} + \beta \int EV(x_{t+1}) P(dx_{t+1} | 0)\}]$$

Ao assumir a "Conditional Independence Assumption" (CI), duas restrições estão envolvidas. Conforme argumenta Rust, x_{t+1} é uma estatística suficiente para ε_{t+1}, o que significa que x_{t+1} transmite qualquer dependência estatística entre ε_t e ε_{t+1}.

Assim, aplicando o Teorema 1:

Se

$$G([u(x,\theta_1)+\beta EV_\theta(x)]|x,\theta_2) \equiv \int_\varepsilon \max_{j\in C(x)} [u(x,j,\theta_1)+\beta EV_\theta(x,j)]q(d\varepsilon|x,\theta_2)$$

Então, $(P_i|x,\theta)$, dado por:

$$(P_i|x,\theta) = G_i([u(x,\theta_1)+\beta EV_\theta(x)]|x,\theta_2)$$

em que G_i denota a derivada parcial de G com respeito à $u(x,i,\theta_1)$ e a função EV_θ é o único ponto fixo para um mapeamento de contração T_θ, $T_\theta(EV_\theta) = EV_\theta$, definido para cada $(x,i) \in \Gamma$ por:

$$EV_\theta(x,i) = \int_y G([u(y,\theta_1)+\beta EV_\theta(y)]|y,\theta_2)p(dy|x,i,\theta_3)$$

Agora, basta passar a função de máximo para avaliar em qual estado se tomará a esperança: naquele em que a troca não é realizada ($-c(x_t,\theta_1)+\varepsilon_{0,t}+\beta\int EV(x_{t+1})P(dx_{t+1}|x_t)$); ou naquele em que ela é realizada ($-R-c(0,\theta)+\varepsilon_{1,t}+\beta\int EV(x_{t+1})P(dx_{t+1}|0)$).

Ou seja:

$$\begin{aligned} EV(x_t) = \mathbb{E}_t[\max\{&-c(x_t,\theta_1)+\varepsilon_{0,t}+\beta\int EV(x_{t+1})P(dx_{t+1}|x_t); \\ &-R-c(0,\theta)+\varepsilon_{1,t}+\beta\int EV(x_{t+1})P(dx_{t+1}|0)\}] \quad \square \end{aligned}$$

h) (*) Prove que a equação anterior é um mapeamento de contração (contraction mapping).

Primeiro, vamos colocar a definiição de contração:

Definição 1 *Seja (X,d) um espaço métrico. Um mapeamento $T: X \to X$ é um mapeamento de contração, ou contração, se existe uma constante c, com $0 \leq c \leq 1$ tal que*

$$d(T(x),T(y)) \leq cd(x,y)$$

para todo $x,y \in X$

Em palavras, uma contração mapeia pontos próximos. Como caso particular, para todo ponto $x \in X$, e qualquer $r > 0$, todos os pontos na bola $B_r(x)$ são mapeados para a bola $B_s(Tx)$, com $r > s$. Se $c < 1$, a contração pode ser chamada de estrita.

Pela CI, temos um mapeamento do espaço $S = \{(x, \varepsilon\}|x \in R^k, \varepsilon \in R^{\#}C(x)\}$ para um espaço contraído $\Gamma = \{(x, i)|x \in R^k, i \in C(x)\}$. Sendo a contração estrita, então existe um único ponto fixo, que, neste caso, é o termo EV_θ no espaço reduzido. □

i) Explique o tradeoff resultante da hipótese de independência condicional (CI). Use um exemplo concreto para explicar o que é descartado (ou deixado de lado). O que esta hipótese "compra" (i.e. ela entrega) para a estimação? Essa hipótese é razoável?

Ao adotar a CI, conforme apontado no item g), qualquer dependência entre os termos não observados pelo econometrista depende apenas de x_t, o que acaba sendo uma hipótese forte, conforme Rust aponta. Todavia, ao adotar tal estratégia, EV_θ deixa de ser função do termo não observado, dispensando que as probabilidades de escolha envolvam uma integração sobre essa função desconhecida. Além disso, a CI implica que EV_θ é um ponto fixo de uma contração sobre o espaço de estado reduzido ($\Gamma = \{(x, i)|x \in R^k, i \in C(x)\}$ ao invés de $S = \{(x, i)|x \in R^k, \varepsilon \in R^{\#C(x)}\}$), o que confere mais um ganho computacional ao evitar o cômputo de mais uma integral para obter EV_θ a partir de V_θ. Ao fazer isso, para o caso dos ônibus, as decisões de troca do motor não observadas pelo econometrista em dois períodos podem depender de outras coisas além da milhagem percorrida. O agente pode, por exemplo, observar as condições das ruas ou a temperatura do ambiente (que pode afetar a temperatura dos componentes do motor e acelerar seu desgaste) para decidir sobre a troca do motor. Pode ainda, por exemplo, optar por direcionar a troca do motor para o períodos em que a demanda por ônibus é menor (como férias) de modo a diminuir eventuais impactos da troca do motor. Todos esses fatores independem da milhagem x_t, afetando a escolha da troca. Apesar de ser uma hipótese forte, parece ser razoável supor que x_t seja o principal fator a influenciar a parcela não observada pelo econometrista. □

Embora equação anterior seja uma contração, ainda é difícil computar sem fazer hipóteses distribucionais adicionais sobre ε. Rust assume que o choque $\varepsilon_{it,t}$ segue uma distribuição valor extremo Tipo I normalizada. Então:

$$\varepsilon_{it,t} \sim EV1\left(\begin{bmatrix}0\\0\end{bmatrix}, \begin{matrix}\frac{\pi^2}{6} & 0\\ 0 & \frac{\pi^2}{6}\end{matrix}\right)$$

j) Por que Rust escolheu uma distribuição padrão para o termo de erro ao invés de adicionar um parâmetro para a sua variância?

Conforme Rust aponta, um valor alto e negativo para $\varepsilon_t(0)$ pode ser interpretado como uma falha não observada de algum componente do ônibus que força para a oficina, enquanto um valor alto e positivo pode ser o reporte do motorista de que o ônibus está bem ajustado. Já $\varepsilon_t(1)$ pode ser associado ao custo associado à troca do motor, em que um valor alto e negativo pode indicar que não há espaço

disponível para reparo ou que não existem motores disponíveis para troca, enquanto um valor alto e positivo indica o inverso. Como o autor não consegue identificar esses tipos de custo sem informação adicional, ele optou por arbitrariamente normalizar a média e variância.

A hipótese de distribuição valor extremo nos permite derivar formas funcionais para $EV(x_t)$, e $P(1|x_t, \theta)$.

k) Discuta a racionalidade por trás da hipótese da distribuição de ε_t. Existem outras alternativas? Escolha uma alternativa e discuta os prós e contras em relação à distribuição valor extremo Tipo I.

Ao escolher essa distribuição, o autor assume implicitamente que $\{x_t^j, \varepsilon_t^j\}$ são independentemente distribuídos entre os diferentes ônibus. Mesmo sendo uma hipótese forte, ela é de menor relevância vis-à-vis sua hipótese de CI. Uma outra possibilidade seria utilizar uma distribuição normal, como no caso de um probit, o que lhe permitiria acomodar possíveis correlações entre o processo decisório envolvendo a troca de motores dos ônibus. Entretanto, isso levaria a complicações na definição do modelo, demandando maiores informações sobre os custos que envolvem o processo decisório de troca do motor.

A última simplificação que Rust faz é discretizar o espaço de x_t divindo a milhagem em 90 intervalos de 5000 milhas. Ele assume que o processo x_t por avançar com pelo menos dois incrementos no tempo, que essencialmente reduz a distribuição de $(x_{t+1} - x_t)$ como uma multinomial com parâmetros $\theta_3 = \{\theta_{30}, \theta_{31}\}$.

l) Escreva $P(x_{t+1}|x_t, i_t, \theta)$.

$$P(x_{t+1}|x_t, i_t, \theta) = \begin{cases} g(x_{t+1} - x_t, \theta) & \text{if } i_t = 0 \\ g(x_{t+1} - 0, \theta) & \text{if } i_t = 1 \end{cases} \quad \square$$

Aqui já se está em posição de estimar o modelo. As únicas duas coisas que não foram especificadas são β e a forma da função custo. Aqui podemos colocar $\beta = 0.9999$[1]. Finalmente, se pode assumir que a função custo é linear:

$$c(x_t, \theta_1) = 0.001\theta_1 x_t$$

m) Observe que não foi incluída uma constante nesta especificação de função custo. O que aconteceria se isto fosse feito? Poderíamos estimar uma constante separadamente? Discuta como isso impacta o resultado final de sua estimativa.

[1] β e R são em geral altamente colineares e portanto de difícil distinção. Para um tratamento rigoroso desta questão veja a seção 3.5 do capítulo escrito por Rust no Handbook of Econometrics (1994). Outra referência útil é o paper de Magnac e Thesmar (Econometrica, 2002).

Segundo Rust, o nível absoluto de c não é identificado uma vez que a subtração de uma constante da função utilidade não afeta as probabilidades de escolha. Assim, o autor busca estimar, ao menos, a mudança nos custos operacionais como função da distância percorrida pelos ônibus, cuja normalização $c(0, \theta_1) = 0$ é utilizada. □

2 Estimação

2.1 Dados

Primeiro é necessário construir a base de dados. A base consiste em oito arquivos ASCII, contendo dados da leitura mensal do odômetro dos 162 ônibus do frota (Madison Metropolitan Bus Company). Estes dados são da operação dos ônibus entre Dezembro de 1974 e Maio de 1985. Cada arquivo corresponde a um modelo/categoria de ônibus da frota da Madison Metro:

1. g870.ASC 36x15 matriz do modelo Grumman 870
2. rt50.ASC 60x4 matriz do modelo Chance RT50
3. t8h203.ASC 81x48 matriz do GMC T8H203
4. a452372.ASC 137x18 matriz do GMC A4523, modelo ano 1972
5. a452374.ASC 137x10 matriz do GMC A4523, modelo ano 1974
6. a530872.ASC 137x18 matriz do GMC A5308, modelo ano 1972
7. a530874.ASC 137x12 matriz do GMC A5308, modelo ano 1974
8. a530875.ASC 128x37 matriz do GMC A5308, modelo ano 1975

Os dados em cada arquivo estão vetorizados em uma única coluna: e.g. g870.ASC é um vetor 540x1 consistindo nas colunas de uma matriz 110x4 que foi empilhada. Comece importando os dados e reorganizando cada vetor em uma matriz. As primeiras 11 "linhas" de cada matriz consiste no metadado:

```
#  Extrair tabela do PDF com a descrição das variáveis

tabulizer::extract_tables("lista2019.pdf",
                          output = "data.frame")[[1]] %>%
  dplyr::bind_rows(dplyr::bind_cols(Row=12,
                                    Field="Monthly odometer reading",
                                    Sample.Entry=4235)) %>%
  xtable::xtable(digits = c(0,0,0,0),
                 caption="Dicionário de variáveis") %>%
  print(include.rownames=F,
                comment=FALSE)
```

Você deve organizar os dados em objetos que são fáceis de manipular para realizar a sua estimação. Em particular, *observe que os dados do odômetro NÃO são zerados quando o motor é substituído. Você*

Tabela 1: Dicionário de variáveis

Row	Field	Sample.Entry
1	Bus ID	5297
2	Month purchased	8
3	Year purchased	75
4	Month of 1st engine replacement	4
5	Year of 1st engine replacement	79
6	Odometer reading at 1st engine replacement	153400
7	Month of 2nd engine replacement	0
8	Year of 2nd engine replacement	0
9	Odometer reading at 2nd engine replacement	0
10	Month odometer data begins	9
11	Year odometer data begins	75
12	Monthly odometer reading	4235

deve realizar esse ajuste.

Importante: Todos os dados estão nos arquivos em anexo, mas Rust roda o principal procedimento para um conjunto restrito de ônibus. Esta restrição é devido a aparente heterogeneidade na matriz de transição do espaço de estado entre os ônibus. Os arquivos que devem ser usados para comparar com os resultados de Rust são 1, 2, 3 e 8.

```
# Carregar a base de dados

#### Ler o dicionário de variáveis
#  Extrair tabela do PDF com a descrição das variáveis
dic <- tabulizer::extract_tables("lista2019.pdf",
                                 output = "data.frame")[[1]]

# Listar arquivos com as bases de dados
bases<- data.frame(arquivo=list.files("rust_data"),
                   nome=gsub(".asc","", list.files("rust_data")),
                   linhas=c(rep(137,4),128,36,60,81),
                   stringsAsFactors = F)

# Looping para carregar as bases
for(b in bases$arquivo){

  # Ler a base de dados
  base <- data.table::fread(paste0("rust_data/",b))

  # Definir o número de linhas da matriz
```

```
  nl<-bases[bases$arquivo==b,]$linhas

  # Definir número de meses existentes na base
  nm<-nl-12+1

  # Criar objeto para receber os dados fixos de cada ônibus
  dados <- c()

  # Criar um objeto para receber as informações mensais
  t<-c()

# Criar um objeto para receber as referências
  ref<-c()

# Iniciar looping para carregar as informações fixas,
# repetindo para o número de meses
for(i in 1:11){

  # Capturar as informações de cada ônibus
  assign(paste0("V",i), unlist(rep(base[seq(i,nrow(base),nl),],nm)))

  # Juntar resultados
  dados <- cbind(dados,get(paste0("V",i)))

}

# Retirar as informações mensais (odômetros)

  for(j in 12:nl){


    # Retirar para cada ônibus
    V12 <- base[seq(j,nrow(base),nl)]

    # Juntar resultados
    t <- rbind(t,V12)
}
```

```r
# Criar um objeto com a referência,
# tendo como base o mês e ano inicial do odômetro


  for(i in seq(10,nrow(base),nl)){

    # Capturar o mês inicial, o ano inicial, definir sempre o primeiro
    # dia de cada mês para transformar em data e criar a sequência de
    # meses conforme o número de meses(nm) disponíveis na base

    r <- data.frame(ref=seq(
      # Define a data
      lubridate::dmy(paste("01",base[i],
                           base[i+1],sep = "/")),
      # Sequência mensal
      by="month",
      # Pelo número de meses
      length.out = nm))

    # Juntar resultados
    ref<- rbind(ref,r)

}

# Juntar a base final
assign(bases[bases$arquivo==b,]$nome,
cbind(dados,V12=t) %>%
  # Organizar por ônibus
  dplyr::arrange(V1) %>%
  # Trazer referência
  dplyr::bind_cols(ref) %>%
  # Ajustar odômetros para ocasião da troca
  dplyr::mutate(V12_adj=case_when(V12.V1>V6&V6>0~V12.V1-V6,
                                  TRUE~V12.V1), # Primeira troca
                V12_adj=case_when(V12.V1>V9&V9>0~V12.V1-V9,
                                  TRUE~V12_adj)) %>% # Segunda troca
  # Agrupar por ônibus
  dplyr::group_by(V1) %>%
```

```
  # Criar indicador de troca
  dplyr::mutate(troca=ifelse(V12_adj>lead(V12_adj),1,0),
                troca=ifelse(is.na(troca)==T,0,troca),
                # Criar variável de estado com
                # intervalos de 5000
                estado=cut(V12_adj,
                           breaks = c(seq(0,max(V12_adj),
                                          by=5000),Inf),
                           labels = FALSE),
                # Calcular a milhagem mensal ajustada
                difV12=V12_adj-lag(V12_adj),
                # Ajustar os casos de troca
                difV12=ifelse(difV12<0,V12_adj,difV12),
                # Definir a mudança de estado nos intervalos
                # [0,5000),[5000,10000),[10000,Inf)
                diffV12_mult=case_when(estado==lag(estado)~1,
                            estado-lag(estado)==1~2,
                            estado-lag(estado)>1~3),
                # Ajustar valor inicial
                diffV12_mult=ifelse(is.na(diffV12_mult)==T,1,
                                    diffV12_mult)) %>%
  # Dropar primeira observação (não há lag)
  na.omit)

# Remover objetos desnecessários
rm(base,dados,r,ref,t,list=ls(pattern = "V\\d"),b,i,j,nl,nm)
}
```

2.2 Computação I: NFXP de Rust

Dada as observações $(\{x_0, i_0\}, \{x_1, i_1\}, ..., \{x_T, i_T\})$ podemos escrever uma função verossimilhança

$$\ell(x_1, ..., x_T, i_1, ..., i_T | x_0, i_0, \theta) = \prod_{t=1}^{T} P(i_t|x_t, \theta).P(x_t|x_{t-1}, i_{t-1}, \theta)$$

Você deve ter fórmulas para $P(i_t|x_t, \theta)$ e $P(x_t|x_{t-1}, i_{t-1}, \theta)$ da Parte 1 da lista.

Rust usa o seguinte procedimento:

Passo 1: Estime $\hat{\theta}_3$ usando

$$\ell^1(x_1, ..., x_T, i_1, ..., i_T | x_0, i_0, \theta) = \prod_{t=1}^{T} P(x_t | x_{t-1}, i_{t-1}, \theta_3)$$

Passo 2: Use as estimativas $\hat{\theta}_3$ como valores consistentes e a estimativa $(\hat{\theta}_1, \hat{R})$ usando

$$\ell^2(x_1, ..., x_T, i_1, ..., i_T | \theta) = \prod_{t=1}^{T} P(i_t | x_t, \theta)$$

Para implementar este método você precisará calcular $EV(x_t)$ para cada valor de θ considerado o algoritmo de máxima verossimilhança. Portanto, o seu código deve ser estruturado para ter um algoritmo de ponto fixo "interno" que calcula $EV(x_t)$ (usando a expressão derivada na parte **k)** e um algoritmo de busca "externo" que maximiza a função de verossimilhança. Um bom ponto de partida pode ser o *payoff* do período ou mesmo o valor de $EV(x_t)$ computado na iteração anterior da estimação.

Passo 3: Usando $(\hat{\theta}_1, \hat{R}, \hat{\theta}_3)$ como um ponto inicial, use a função de verossimilhança completa para obter uma estimativa eficiente dos parâmetros.

n) Estime (θ_1, R, θ_3) usando a metodologia de Rust de maximização da função de verossimilhança na equação 2 (ou uma função log-verossimilhança equivalente)[2]. Você deve comparar os números que estimou com os obtidos por Rust no seu paper (a tabela relevante é Table IX).

```
# Definição da função custo e a função escolha (Passo 2)
source("cmiope.R",encoding ="UTF-8")

# Algoritmo do mapeamento de contração
# (gera a escolha de probabilidade "forward-looking")
source("cont_map.R",encoding ="UTF-8")

# 1) Modelo Logit de utilidade dinâmica
source("dinam_logit.R",encoding ="UTF-8")


# Definir base: Grupo 1,2,3
```

[2]Você pode cair em um problema de estimação porque sua linguagem de programação aproxima números bem pequenos para zero. Este é um problema quando estiver calculando (como você verá) $log(exp(x_1) + exp(x_2))$ onde tanto x_1 ou x_2 (ou ambos) são números negativos bem grandes. Nesse caso, se sua linguagem de programação soluciona $exp(x_1) = 0$, então $log(exp(x_1) + exp(x_2)) = x_2$, que é errado. Você pode encontrar uma descrição detalhada deste problema em https://lips.cs.princeton.edu/computing-log-sum-exp/. A solução consiste essencialmente em tirar $exp(x_1)$ da soma do log:

$$log(exp(x_1) + exp(x_2)) = log(exp(x_1).(1 + exp(x_2 - x_1))) = x_1 + log((1 + exp(x_2 - x_1)))$$

```r
d1 = rbind(g870[,c("V1","troca","V12_adj","estado",
                   "difV12","diffV12_mult")],
           rt50[,c("V1","troca","V12_adj","estado",
                   "difV12","diffV12_mult")],
           t8h203[,c("V1","troca","V12_adj","estado",
                     "difV12","diffV12_mult")])

# Grupo 4
d2= rbind(a530875[,c("V1","troca","V12_adj","estado",
                     "difV12","diffV12_mult")])

# Grupo 1,2,3 e 4
d3= rbind(g870[,c("V1","troca","V12_adj","estado",
                   "difV12","diffV12_mult")],
           rt50[,c("V1","troca","V12_adj","estado",
                   "difV12","diffV12_mult")],
           t8h203[,c("V1","troca","V12_adj","estado",
                     "difV12","diffV12_mult")],
           a530875[,c("V1","troca","V12_adj","estado",
                     "difV12","diffV12_mult")])

# Objeto para receber os resultados

resultados <- c()

erro <- c()

# Calcular para cada base
for(i in c("d1","d2","d3")){

  # Defifnir a base do loop
  data <- get(i)

  # Definir o label do loop
if(i=="d1"){
  label="Grupo 1, 2 e 3"
} else{if(i=="d2"){
  label="Grupo 4"} else{
```

```
    label="Grupo 1, 2, 3 e 4"
  }
}
  ####################
  ###### Passo 1 #####
  ####################

# Verificar a probabilidade empírica da mudança em cada estado

p <- data %>%
  # Analisar as mudanças de estado
  dplyr::mutate(p=case_when(estado==lag(estado)~"p_x0",
                            estado-lag(estado)==1~"p_x1",
                            estado-lag(estado)>1~"p_x2"),
                p=ifelse(is.na(p)==T,"p_x0",p)) %>%
  # Agrupar pelas quebras
  dplyr::group_by(p) %>%
  # Contar a quantidade de observações nas quebras
  dplyr::summarise(n=n()) %>%
  # Desagrupar
  dplyr::ungroup() %>%
  # Calcular a proporção
  dplyr::mutate(n=n/sum(n))
# Armazenar proporção em um objeto
p <- p$n

# p_x0 é a chance de manutenção na mesma faixa de milhagem,
# p_x1 é a chance de passa para o status 2,
# enquanto p_x1 chance de estar no último status

# Decisão do agente

# Variável iniciais
# O custo de troca, os parâmetros de manutenção da função custo,
# e a taxa de desconto são definidas aqui.
# A taxa de desconto de beta era pra ser = 0.9999
# Por questões de convergência, consegui somente com 0.91
# Os tetas iniciais foram definidos de acordo com os resultados
```

```r
# da tabela IX do paper

if(i=="d1"){
rc=20
theta1_1=5
beta=0.91
} else{

  if(i=="d2"){
    rc=20
    theta1_1=2
    beta=0.91} else{
      rc=20
      theta1_1=2
      beta=0.91
    }
  }

# Definir parâmetros iniciais
params_lin = c(rc,theta1_1)
p = p
S = 90

#####################
###### Passo 2 #####
#####################

# Rodar modelo
out = contraction_mapping(S=S,
                          p=p,
                          FCM=lin_cost,
                          params=params_lin,
                          beta = beta)

# Guardar resultados em um objeto
lin_forward=out$CP_forward
lin_miope=out$CP_miope
pescolha = lin_forward[,1]
```

```r
# Definir base com resultados
ggdat1 = data.frame(Regra=c(rep("Forward-Looking",nrow(lin_forward)),
                            rep("Míope",nrow(lin_forward))),
                    pMaint=c(lin_forward[,1],lin_miope[,1]),
                    State=rep(1:S,times=2))

# Fazer o gráfico
assign(paste0("g_",i),
ggplot(ggdat1,aes(y=pMaint,x=State,color=Regra))+
  geom_line(lwd=1)+
  theme_bw(12)+
  scale_x_continuous(breaks = seq(0,90,by=10))+
  scale_y_continuous(labels = scales::percent)+
  labs(x="Estado",
       y="Prob.",
       caption = label))


# limites
bounds = c(1e-9, Inf)

####################
###### Passo 3 #####
####################

lin_fit = optim(par=params_lin,fn=dynamiclogit,method=c("L-BFGS-B"),
                lower=bounds[1],upper=bounds[2],
                data=data,S=S,p=p,FCM=lin_cost,
                control=list(fnscale=1),hessian = T)
# Retornar os parâmetros obtidos com a função
loglike =  lin_fit$value
fit_params = lin_fit$par

# Computar erro-padrão
EP <-sqrt(diag(solve(lin_fit$hessian)))

# Armazenar resultados no objeto
er <- data.frame(Grupo=label,
```

```
                  EP_RC=EP[1],
                  EP_T1=EP[2])

# Agregar resultados
erro <- rbind(erro,er)

# Armazenar resultados no objeto
r <- data.frame(Grupo=label,
                LL=loglike*-1,
                RC=fit_params[1],
                T1=fit_params[2],
                T30=p[1],
                T31=p[2])

# Agregar resultados
resultados <- rbind(resultados,r)
}
```

```
## Convergência atingida em  212  iteraçõesConvergência atingida em  202  iteraçõesCo
```

Os resultados podem ser observados na Tabela 2

```
resultados %>%
  xtable::xtable(digits = c(0,rep(3,ncol(resultados))),
                 caption="Estimativas para a questão n)",
                 label="tab:2") %>%
  print(include.rownames=F,
        comment=FALSE)
```

Tabela 2: Estimativas para a questão n)

Grupo	LL	RC	T1	T30	T31
Grupo 1, 2 e 3	-137.141	7.274	4.193	0.318	0.671
Grupo 4	-169.110	6.503	2.399	0.401	0.586
Grupo 1, 2, 3 e 4	-315.070	6.307	2.254	0.362	0.626

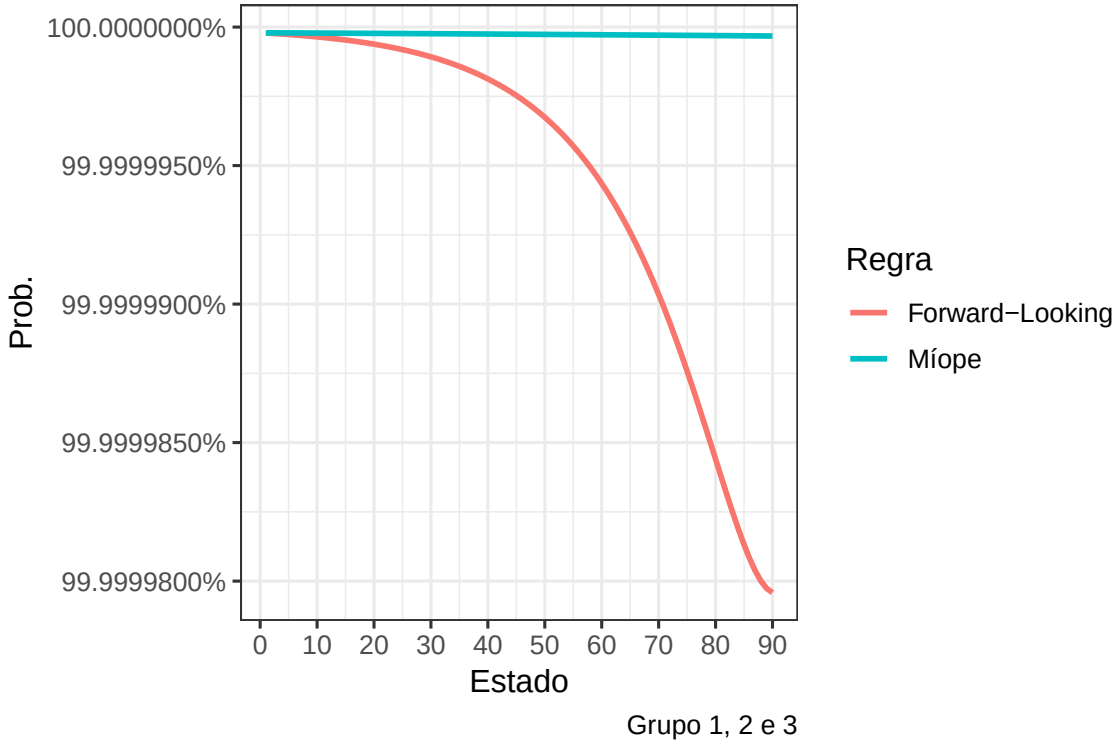
Como não foi possível utilizar o mesmo valor de β utilizado no paper, os resultados infelizmente não são diretamente comparáveis. Para RC, todos os valores estimados ficaram abaixo dos valores estimados por Rust (resultado do menor valor para taxa de desconto utilizado), porém a tendência declinante entre os conjuntos de dados é a mesma. Os valores de T1 (θ_1) são mais parecidos com aqueles calculados por Rust, mas também um pouco inferiores para os três conjuntos de dados. Por fim, os valroes de T30 (θ_{30}) e T31 (θ_{31}) são muito próximo daquels observados por Rust, alguns sendo

ligeiramente superiores ou inferiores.

Abaixo temos os gráficos da troca para $\beta = 0$ (míope) e $\beta = 1$ (foward-looking), para cada um dos grupos, nas Figuras 1, 2 e ref{fig:3}. □

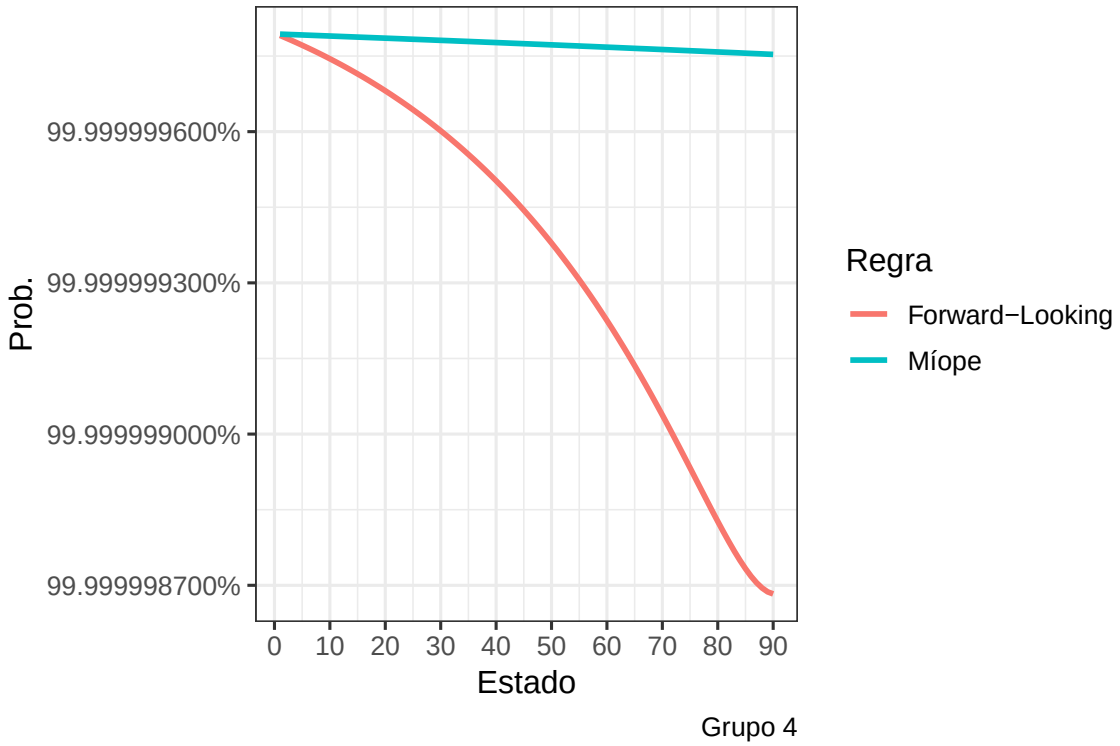
```
g_d1
```

Figura 1: Probabilidade de troca, Grupo 1, 2 e 3.



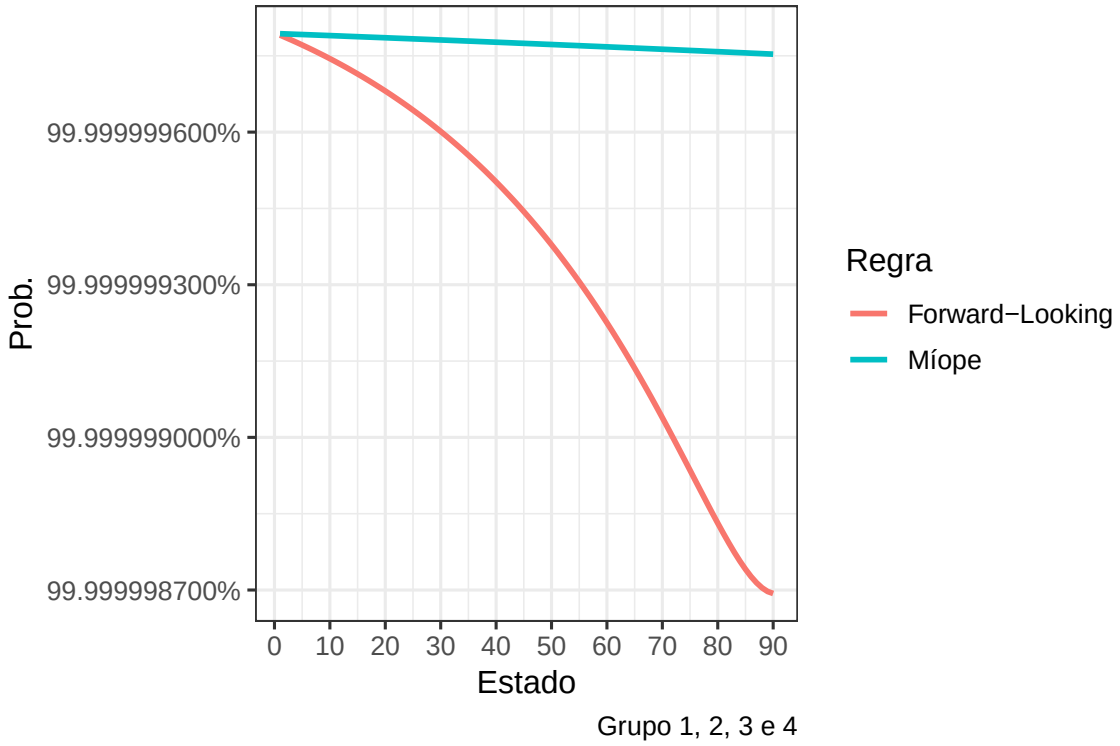

```
g_d2
```

Figura 2: Probabilidade de troca, Grupo 4.




```
g_d3
```

Figura 3: Probabilidade de troca, Grupo 1, 2, 3 e 4.



o) Explique como você pode calcular os erros padrão para estes coeficientes.

Os erros-padrão para esses coeficientes podem ser calculadas a partir da hessiana da matriz derivada das estimativas da função verossimilhança, realizado no momento do passo 3 descrito anteriormente. □

p) (*) Calcule o erro padrão usando a metodologia que você descreveu no item o).

Na tabela 3 temos os valores calculados na questão n) (ver código da questão referida).

```
erro %>%
  xtable::xtable(digits = c(0,rep(3,ncol(erro))),
                 caption="Estimativas do erro-padrão de RC e T1",
                 label="tab:3") %>%
  print(include.rownames=F,
        comment=FALSE)
```

Tabela 3: Estimativas do erro-padrão de RC e T1

Grupo	EP_RC	EP_T1
Grupo 1, 2 e 3	0.589	1.018
Grupo 4	0.388	0.419
Grupo 1, 2, 3 e 4	0.273	0.320

Novamente, os valores não são diretamente comparáveis com a Tabela IX por questões metodológicas distintas. □

q) (*) Rust estimou diferentes versões do seu modelo, variando (i) a função custo, (ii) o parâmetro β e (iii) o número de valores possíveis para a variável de estado x_t. Ele relata alguns resultados para todas parametrizações diferentes do modelo. Baseado apenas no que ele apresenta no paper, argumente argumente quais destas três hipóteses possuem maior consequência. Descreva e motive uma forma de relaxar a hipótese (cabe a você desenvolver isto e fique a vontade para usar uma das expansões apresentadas por Rust). Refaça a estimação com esta hipótese relaxada. Comente sobre qualquer diferença relevante nos resultados.

Primeiramente, com relação ao β, o autor argumenta pouca sensibilidade do parâmetro a mudanças de valores, por exemplo, de 0.9999 ou 0.98. Rust argumenta que há colinearidade entre o custo de substituição do motor e a taxa de desconto intertemporal, uma vez que ambos capturam efeitos similares no comportamento do agente. Com relação aos formato da função custo, o autor aponta que as funções com termos quadráticos e cúbicos mostram um comportamento extremo, muito provavelmente devido às poucas observações com altos valores de milhagem, tornando as estimativas nas caudas das distribuições mais erráticas. Assim, as funções lineares e com raíz quadrada, após análise conjunta com Zurcher, parecem ser as mais coerentes com o comportamento real dos dados. Por fim, ao mudar o discretização para quase o dobro de intervalos, de 5000 para 2571, o autor argumenta não haver

importantes alterações nas conclusões. O que passa a ocorrer é que, com o afinamento dos intervalos, aumenta-se o número de observações nos intervalos de baixa probabilidade. O autor também informa ter realizado testes com intervalos fixos com dimensão 45, havendo pouca alteração nos coeficientes estimados. Diante do exposto, a função custo parece ser a mais importante, a qual necessitou, inclusive, uma avaliação do agente para a tomada de decisão. Assim sendo, vamos verificar um terceiro caminho: aumentar os intervalos, cortando os grids pela metade e dobrando a dimensão dos estados, além de alterar a função custo para uma função logarítmica.

```
# Preparar a base com estados em intervalos de 2.500 milhas

# Looping para carregar as bases
for(b in bases$arquivo){

  # Ler a base de dados
  base <- data.table::fread(paste0("rust_data/",b))

  # Definir o número de linhas da matriz
  nl<-bases[bases$arquivo==b,]$linhas

  # Definir número de meses existentes na base
  nm<-nl-12+1

  # Criar objeto para receber os dados fixos de cada ônibus
  dados <- c()

  # Criar um objeto para receber as informações mensais
  t<-c()

# Criar um objeto para receber as referências
  ref<-c()

# Iniciar looping para carregar as informações fixas,
# repetindo para o número de meses
for(i in 1:11){

  # Capturar as informações de cada ônibus
  assign(paste0("V",i), unlist(rep(base[seq(i,nrow(base),nl),],nm)))

  # Juntar resultados
```

```
  dados <- cbind(dados,get(paste0("V",i)))

}

# Retirar as informações mensais (odômetros)

  for(j in 12:nl){


    # Retirar para cada ônibus
    V12 <- base[seq(j,nrow(base),nl)]

    # Juntar resultados
    t <- rbind(t,V12)
}

# Criar um objeto com a referência,
# tendo como base o mês e ano inicial do odômetro


  for(i in seq(10,nrow(base),nl)){

    # Capturar o mês inicial, o ano inicial, definir sempre o primeiro
    # dia de cada mês para transformar em data e criar a sequência de
    # meses conforme o número de meses(nm) disponíveis na base

    r <- data.frame(ref=seq(
      # Define a data
      lubridate::dmy(paste("01",base[i],
                           base[i+1],sep = "/")),
      # Sequência mensal
      by="month",
      # Pelo número de meses
      length.out = nm))

    # Juntar resultados
    ref<- rbind(ref,r)
```

```
}

# Juntar a base final
assign(bases[bases$arquivo==b,]$nome,
cbind(dados,V12=t) %>%
  # Organizar por ônibus
  dplyr::arrange(V1) %>%
  # Trazer referência
  dplyr::bind_cols(ref) %>%
  # Ajustar odômetros para ocasião da troca
  dplyr::mutate(V12_adj=case_when(V12.V1>V6&V6>0~V12.V1-V6,
                                  TRUE~V12.V1), # Primeira troca
                V12_adj=case_when(V12.V1>V9&V9>0~V12.V1-V9,
                                  TRUE~V12_adj)) %>% # Segunda troca
  # Agrupar por ônibus
  dplyr::group_by(V1) %>%
  # Criar indicador de troca
  dplyr::mutate(troca=ifelse(V12_adj>lead(V12_adj),1,0),
                troca=ifelse(is.na(troca)==T,0,troca),
                estado=cut(V12_adj,
                           breaks = c(seq(0,max(V12_adj), by=2500),Inf),
                           labels = FALSE),
                difV12=V12_adj-lag(V12_adj),
                difV12=ifelse(difV12<0,V12_adj,difV12),
                diffV12_mult=case_when(estado==lag(estado)~1,
                            estado-lag(estado)==1~2,
                            estado-lag(estado)==2~3,
                            estado-lag(estado)==3~4,
                            estado-lag(estado)==4~5,
                            estado-lag(estado)>4~6,),
                diffV12_mult=ifelse(is.na(diffV12_mult)==T,
                                    1,
                                    diffV12_mult)) %>%
  na.omit)  # Ajustar primeiro valor


# Remover objetos desnecessários
rm(base,dados,r,ref,t,list=ls(pattern = "V\\d"),b,i,j,nl,nm)
```

```
}

# Salvar as novas bases

# Grupo 1,2,3
d4 = rbind(g870[,c("V1","troca","V12_adj","estado",
                   "difV12","diffV12_mult")],
           rt50[,c("V1","troca","V12_adj","estado",
                   "difV12","diffV12_mult")],
           t8h203[,c("V1","troca","V12_adj","estado",
                     "difV12","diffV12_mult")])

#Grupo 4
d5= rbind(a530875[,c("V1","troca","V12_adj","estado",
                     "difV12","diffV12_mult")])

# Grupo 1,2,3 e 4
d6= rbind(g870[,c("V1","troca","V12_adj","estado",
                   "difV12","diffV12_mult")],
           rt50[,c("V1","troca","V12_adj","estado",
                   "difV12","diffV12_mult")],
           t8h203[,c("V1","troca","V12_adj","estado",
                     "difV12","diffV12_mult")],
           a530875[,c("V1","troca","V12_adj","estado",
                     "difV12","diffV12_mult")])


# Objeto para receber os resultados

resultados <- c()

# Calcular para cada base
for(i in c("d4","d5","d6")){

data <- get(i)

if(i=="d4"){
  label="Grupo 1, 2 e 3"
```

```
} else{if(i=="d5"){
  label="Grupo 4"} else{

    label="Grupo 1, 2, 3 e 4"
  }
}



# Verificar a probabilidade empírica da mudança em cada estado
p <- data %>%
  # Analisar as mudanças de estado
  # Agrupar pelas quebras
  dplyr::group_by(diffV12_mult) %>%
  # Contar a quantidade de observações nas quebras
  dplyr::summarise(n=n()) %>%
  # Desagrupar
  dplyr::ungroup() %>%
  # Calcular a proporção
  dplyr::mutate(n=n/sum(n))

# Armazenar proporção em um objeto
p <- p$n

# Decisão do agente

# Variável iniciais
# O custo de troca, os parâmetros de manutenção da função custo,
# e a taxa de desconto são definidas aqui.
# A taxa de desconto de beta era pra ser = 0.9999
# Por questões de convergência, consegui somente com 0.95

if(i=="d4"){
rc=11
theta1_1=5
theta1_2=0.01
beta=0.91
} else{
```

```
  if(i=="d5"){
    rc=10
    theta1_1=2
    theta1_2=0.01
    beta=0.91} else{

      rc=10
      theta1_1=3
      theta1_2=0.01
      beta=0.91
    }
  }

# Definir parâmetros iniciais
params_log = c(rc,theta1_1,theta1_2)
p = p
S = 180

# Rodar modelo
out = contraction_mapping(S=S,
                          p=p,
                          # Mudando a função de custo
                          FCM=log_cost,
                          params=params_log,
                          beta = beta)

lin_forward=out$CP_forward
lin_miope=out$CP_miope
pescolha = lin_forward[,1]

ggdat1 = data.frame(Regra=c(rep("Forward-Looking",nrow(lin_forward)),
                                  rep("Míope",nrow(lin_forward))),
                    pMaint=c(lin_forward[,1],lin_miope[,1]),
                    State=rep(1:S,times=2))

# Fazer o gráfico
assign(paste0("g_",i),
ggplot(ggdat1,aes(y=pMaint,x=State,color=Regra))+
```

```
  geom_line(lwd=1)+
  theme_bw(12)+
  scale_x_continuous(breaks = seq(0,90,by=10))+
  scale_y_continuous(labels = scales::percent)+
  labs(x="Estado",
       y="Prob.",
       caption = label))

# Estimação aplicação à função custo

# limites
bounds = c(1e-9, Inf)



# Aplicação ao custo linear
log_fit = optim(par=params_log,fn=dynamiclogit,method=c("L-BFGS-B"),
                lower=bounds[1],upper=bounds[2],
                data=data,S=S,p=p,FCM=log_cost,control=list(fnscale=1))

# Retornar os parâmetros obtidos com a função
loglike =  log_fit$value
fit_params = log_fit$par

# Computar erro-padrão
EP <-sqrt(diag(solve(lin_fit$hessian)))

# Armazenar resultados no objeto
er <- data.frame(Grupo=label,
                 EP_RC=EP[1],
                 EP_T1=EP[2])

# Agregar resultados
erro <- rbind(erro,er)

# Armazenar resultados no objeto
r <- data.frame(Grupo=label,
                LL=loglike*-1,
                RC=fit_params[1],
```

```
                T11=fit_params[2],
                T12=fit_params[3],
                T30=p[1],
                T31=p[2],
                T32=p[3],
                T34=p[4],
                T35=p[5])

# Agregar resultados
resultados <- rbind(resultados,r)

}
```

```
## Convergência atingida em  273  iteraçõesConvergência atingida em  248  iteraçõesCo
```

```
resultados %>%
  xtable::xtable(digits = c(0,rep(3,ncol(resultados))),
                 caption="Estimativas para o modelo modificado:
                 Grid: 2.500, S: 180 e função custo logarítmica",
                 label="tab:4") %>%
  print(include.rownames=F,
        comment=FALSE)
```

Tabela 4: Estimativas para o modelo modificado: Grid: 2.500, S: 180 e função custo logarítmica

Grupo	LL	RC	T11	T12	T30	T31	T32	T34	T35
Grupo 1, 2 e 3	-166.892	26.073	7.203	0.004	0.096	0.420	0.463	0.021	0.000
Grupo 4	-234.113	31.547	5.941	0.007	0.118	0.558	0.300	0.021	0.001
Grupo 1, 2, 3 e 4	-447.602	29.404	6.229	0.005	0.108	0.493	0.377	0.021	0.001

Conforme resultados da Tabela 4, observa-se que tanto o custo de troca quanto os parâmetros para custo de manutenção são mais elevados que aqueles estimados anteriormente (ainda que estes últimos não sejam comparáveis com os anteriores). Além disso, com o aumento dos grids, tem-se que as probabilidades estimadas de mudança para os estados mais extremos são mais baixas. Também verifica-se que a estimativa para o custo de troca é mais elevada para o segundo grupo de dados, ao passo que o parâmetro de custo de manutenção T1 é menor. Na amostra completa, os custos de combinam, elevando o custo de troca em relação à estimativa do primeiro conjunto de dados e diminuindo em relação ao custo de manutenção. Assim, devido ao aumento observado nos parâmetros de custo de troca, verifica-se uma aparente sensibilidade em relação a forma funcional utilizada. □

2.3 Computação II: método CCP

O método de estimação montado por Rust é "time-consuming" porque requer que você compute um ponto fixo para cada "guess" de θ. Hotz e Miller propõem um método alternativo.

r) Estime $\hat{F}(x_{t+1}|x_t, i_t)$ a partir dos dados. Você pode usar qualquer método (parametric, non-parametric, etc.). Justifique sua escolha.

Vamos estimar esses parâmetros com um logit multinomial paramétrico, por simplicidade, uma vez que sua forma é fechada e de fácil computação.

```
# Objeto para receber os dados
resultados <- c()

# Iniciar o loop de estimativa para cada conjunto de dados
for(i in c("d1","d2","d3")){

data <- get(i)

if(i=="d1"){
  label="Grupo 1, 2 e 3"
} else{if(i=="d2"){
  label="Grupo 4"} else{
    label="Grupo 1, 2, 3 e 4"
  }
}

# Rodar o logit multinomial
l_m <- nnet::multinom(lead(diffV12_mult)~diffV12_mult-1,
                      family = "binomial",data = data)

# Armazenar as estimativas
est = l_m$fitted.values

# Computar a probabilidade
p <- c(mean(est[,1]),mean(est[,2]),mean(est[,3]))

# Armazenar resultados
r <- data.frame(Grupo=label,
                p1=mean(est[,1]),
                p2=mean(est[,2]),
```

```
                p3=mean(est[,3]))

# Agregar resultados
resultados <- rbind(resultados,r)

}
```

```
# weights:  6 (2 variable)
initial  value 4243.939271
iter  10 value 2656.830413
final  value 2656.829742
converged
# weights:  6 (2 variable)
initial  value 4714.145331
iter  10 value 3241.141811
final  value 3241.141714
converged
# weights:  6 (2 variable)
initial  value 8959.183214
iter  10 value 5947.212685
final  value 5947.195630
converged
```

```
# Tabela
resultados %>%
  xtable::xtable(digits = c(0,3,3,3,3),
                 caption="Estimativas do Logit multinomial",
                 label="tab:5") %>%
  print(include.rownames=F,
        comment=FALSE)
```

Tabela 5: Estimativas do Logit multinomial

Grupo	p1	p2	p3
Grupo 1, 2 e 3	0.332	0.654	0.013
Grupo 4	0.437	0.545	0.018
Grupo 1, 2, 3 e 4	0.387	0.598	0.016

□

s) Estime $\hat{P}(i_t = 1|x_t)$ a partir dos dados. Novamente justifique sua escolha de método.

Vamos estimar esse parâmetro com um logit, novamente por simplicidade

```r
# Objeto para receber os dados
resultados <- c()

# Iniciar loop para cada conjunto de dados
for(i in c("d1","d2","d3")){

data <- get(i)

if(i=="d1"){
  label="Grupo 1, 2 e 3"
} else{if(i=="d2"){
  label="Grupo 4"} else{
    label="Grupo 1, 2, 3 e 4"
  }
}

# Rodar o logit
glmout=glm(troca~difV12-1,family='binomial',data=data)
# Recuperar o coeficiente
coef=glmout$coef
# Computar estimaticas
est = exp(-(cbind(data$difV12)%*%coef))
# Calcular probabilidade
prob0 = 1/(1+est)

# Armazenar resultados
r <- data.frame(Grupo=label,
                p_troca=mean(prob0),
                # Probabilidade complementar
                p_mant=mean(1-prob0))

# Agregar resultados
resultados <- rbind(resultados,r)

}
```

```r
# Exibir tabela
resultados %>%
  xtable::xtable(digits = c(0,3,3,3),
```

```
                caption="Estimativas",
                label="tab:6") %>%
  print(include.rownames=F,
        comment=FALSE)
```

Tabela 6: Estimativas

Grupo	p_troca	p_mant
Grupo 1, 2 e 3	0.040	0.960
Grupo 4	0.021	0.979
Grupo 1, 2, 3 e 4	0.031	0.969

□

Agora faça

$$\tilde{V}(i_t, x_t; \theta) \equiv \mathbb{E}_{\varepsilon_{i_t,t}}[V(i_t, x_t, \varepsilon_t; \theta)]$$

Uma vez que o termo de erro é média zero, temos

$$\tilde{V}(i_t, x_t; \theta) \equiv -c(x_t, i_t; \theta_1) + \beta \mathbb{E}_{\varepsilon_{i_{t+1},t+1}} \left[u(x_{t+1}, i_{t+1}; \theta) + \varepsilon_{i_{t+1},t+1} + \beta \mathbb{E}_{\varepsilon_{i_{t+2},t+2}}[\cdots] \right]$$

tal que as variáveis são distribuídas como

$$x_{t+1} \sim \hat{F}(.|x_t, i_t) i_{t+1} \sim \hat{P}(.|x_{t+1}) x_{t+2} \sim \hat{F}(.|x_{t+1}, i_{t+1})$$

e as expectativas são condicional, significando que elas são tomadas mantendo a história fixa (e conhecida). As propriedades da distribuição valor extremo Tipo I nos dizem que

$$\mathbb{E}[\varepsilon_{i_t,t}|i_t, x_t] = \gamma - log(P(i_t|x_t))$$

tal que γ é a constante de Euler.

t) Use um argumento recursivo para escrever $\tilde{V}(i_t, x_t; \theta)$ como uma soma infinitamente descontada (infinite discounted sum).

Para $\tilde{V}(i_t = 1, x_t; \theta)$:

$$\begin{aligned}\tilde{V}(i_t = 1, x_t; \theta) = u(x, i_t = 1; \theta) + \beta \mathbb{E}_{x'|x,i=1} \mathbb{E}_{i'|x'} \mathbb{E}_{\varepsilon'|i',x'} [u(x', i'; \theta) + \varepsilon' \\ + \beta \mathbb{E}_{x''|x',i'} \mathbb{E}_{i''|x''} \mathbb{E}_{\varepsilon'|i'',x''} [u(x'', i''; \theta) + \varepsilon'' + \beta \dots]]\end{aligned}$$

De maneira análoga, para $\tilde{V}(i_t = 0, x_t; \theta)$*:*

$$\tilde{V}(i_t = 0, x_t; \theta) = u(x, i_t = 0; \theta) + \beta \mathbb{E}_{x'|x,i=1} \mathbb{E}_{i'|x'} \mathbb{E}_{\varepsilon'|i',x'} [u(x', i'; \theta) + \varepsilon' + \beta \mathbb{E}_{x''|x',i'} \mathbb{E}_{i''|x''} \mathbb{E}_{\varepsilon'|i'',x''} [u(x'', i''; \theta) + \varepsilon'' + \beta \dots]]$$

Que pode ser simulada por:

$$\tilde{V}(i_t, x_t; \theta) \simeq \frac{1}{S} \sum_s \Bigg[u(x_t, i_t; \theta) + \gamma - log(\hat{P}(i_t|x_t)) + \beta \Big[u(x'^s, i'^s; \theta) + \gamma - log(\hat{P}(i_t'^s|x_t'^s)) + \beta \Big[u(x''^s, i''^s; \theta) + \gamma - log(\hat{P}(i_t''^s|x_t''^s)) + \beta \dots \Big] \Big] \Bigg]$$

Em que $x'^s \sim \hat{G}(.|x, i)$,$i'^s \sim \hat{p}(.|x'^s)$, $x''^s \sim \hat{G}(.|x'^s, i'^s)$ *e assim sucessivamente.* □

Você deve ser capaz de calcular a expressão que você derivou para $\tilde{V}(i_t, x_t; \theta)$ usando simulação numérica. Faça $\{(x_t^s, i_s^t)\}_{t=1,s=1}^{T,S}$ serem os valores simulados de x_t e i_t. Então você pode calcular $\tilde{V}^s(i_t^s, x_t^s; \theta)$ e obter uma estimativa consistente de

$$\tilde{V}^{sim}(i, x; \theta) = \frac{1}{S} \sum_{s=1}^{S} \tilde{V}^s(i_t^s, x_t^s; \theta)$$

Seu $\tilde{V}^{sim}$ deve ser uma função de q que é essencialmente calculado para cada "guess". Observe que este passo deve fazer o estimador mais rápido (uma vez que você não precisa computar a iteração de ponto fixo toda vez que tiver um novo "guess" de θ. Entretanto, o quão rápido será o seu estimador depende do quão "objetivo" (esperto) será a sua escolha de simulação. Em particular tenha em mente o seguinte: (i) é mais rápido sortear um número de uma matriz existente do que simular um número; e (ii) seus sorteios de (x_t^s, i_t^s) não dependem dos parâmetros que você está estimando.

Para concluir o procedimento, relembre do começo que usando a hipótese valor extremo T1

$$\tilde{P}(i_t = 1|x_t; \theta) = \frac{exp(\tilde{V}(x_t, i_t = 1; \theta))}{exp(\tilde{V}(x_t, i_t = 0; \theta)) + exp(\tilde{V}(x_t, i_t = 1; \theta))}$$

Isto fornece um $\tilde{P}$ simulado para cada "guess" de θ e agora o que falta fazer é encontrar a condição de momento para a estimação (basicamente qualquer coisa que é baseado em $\|\tilde{P} - \hat{P}\| = 0$ irá funcionar, embora você pode pensar se irá querer uma matriz de pesos para tornar a estimação mais eficiente).

u) Estime θ_1 **e** R **usando a metodologia Hotz-Miller. Compare os resultados com aqueles**

obtidos usando o método de Rust. Comente as diferenças.

```
# Fonte:
# https://github.com/waynejtaylor/Single-Agent-Dynamic-Choice/blob/
# master/Import%20Data%20and%20Estimate.R



#  Programa para estimar o modelo de Rust, Econometrica 1987,
#  usando o algorítmo Nested Pseudo Likelihood (NPL) de
#  Aguirregabiria e Mira (Econometrica, 2002).

#  Funções auxiliares
source('npl_sing.R',encoding ="UTF-8")
source('clogit.R',encoding ="UTF-8")



# Definindo as constantes

# Nome dos parâmetros estruturais
namespar = c("ReplaceC","MaintenC")

# Número de variáveis de estado
kvarx = 1

# Número de alternativas
jchoice = 2

# Número de grids
ncelx = 90

# Objeto para receber os resultados
resultados <- c()

# Calcular os resultados para cada base

# Iniciar o loop
for(d in c("d1","d2","d3")){

  # Definir a base
```

```
data <- get(d)

# Definir o label
if(i=="d1"){
  label="Grupo 1, 2 e 3"
} else{if(i=="d2"){
  label="Grupo 4"} else{

    label="Grupo 1, 2, 3 e 4"
  }
}

# Definir o beta

if(d=="d1"){
beta=0.9999
} else{

  if(i=="d2"){
    beta=0.9999} else{

    beta=0.99
    }
  }

# Objeto com a decisão de troca
aobs = data$troca

# Ajuste da troca (para começar em 1)
indobsa = aobs+1

# Custo de manutenção em cada estado
xval = 1:ncelx

# Indicados do estado (deve começar em 1)
indobsx = data$estado

# 3. Função utilidade
```

```r
# Com manutenção (zmat1),
# custo da troca = 0 e custo de manutenção = -xval
zmat1= cbind(0,-xval)

# Se troca (zmat2), custo da troca = 1
# e custo da manutenção = 0
zmat2= cbind(-1,rep(0,ncelx))
# Juntar matrizes
zmat=cbind(zmat1,zmat2)
# Remover objetos temporários
rm(zmat1,zmat2)

# Outra alternativa
# zmat1= cbind(1,-xval)
# zmat2= cbind(0,rep(0,ncelx))
# zmat=cbind(zmat1,zmat2)
# rm(zmat1,zmat2)

# 4. Estimativa das probabilidades de transição

# Definir a matriz
fmat1 = matrix(0,ncelx,ncelx)
# Definir as probabilidades
# Verificar a probabilidade empírica da mudança em cada estado
l_m <- nnet::multinom(lead(diffV12_mult)~diffV12_mult-1,
                      family = "binomial",data = data)


est = l_m$fitted.values

p <- c(mean(est[,1]),mean(est[,2]),mean(est[,3]))

# Definir o tamanho do vetor
lp = length(p)
# Inicir o loop
for(i in 1:ncelx){
    for(j in 1:lp){
      if((i+j-1)<ncelx)  fmat1[i,i+j-1] = p[j]
```

```
      if((i+j-1)==ncelx) fmat1[i,i+j-1] = sum(p[j:lp])
    }
  }
  fmat2 = cbind(1,matrix(0,ncelx,ncelx-1))
  fmat=cbind(fmat1,fmat2)


#**********************************
# 5. Probabilidades iniciais
#**********************************

# Fazer com base numa polinomial de indobsx
xprob0=cbind(indobsx,indobsx^2,indobsx^3)
# Rodar logit
glmout=glm(aobs~xprob0-1,family='binomial')
# Recuperar coeficientes
coef=glmout$coef
# Definir estimativa
est = exp(-(cbind(xval,xval^2,xval^3)%*%coef))
# Calcular probabilidade de troca
prob0 = 1/(1+est)
# Calcular o complementar (manter) e guardar no objeto
prob0 = cbind(1-prob0,prob0)

# 6. Estimativa estrutural

# Rodar o NPL
out = npl_sing(indobsa,indobsx,zmat,prob0,beta,fmat,namespar)

# Recuperar LL
LL = out[[length(out)]][[4]]

# Recuperar parâmetro RC
RC <- out[[length(out)]][[1]][1]

# Calcular SE
RC_SE <- sqrt(out[[length(out)]][[2]][1])
```

```r
# Recuperar parâmetro T1
T1 <- out[[length(out)]][[1]][2]

# Cacular SE
T1_SE <- sqrt(out[[length(out)]][[2]][4])

# Juntar resultados
r <- data.frame(Grupo=label,
                LL=LL,
                RC=RC,
                RC_SE=RC_SE,
                T1=T1*1000,
                T1_SE=T1_SE*1000)

# Juntar com resultado do grupo anterior
resultados <- rbind(resultados,r)

}
```

```
# weights:  6 (2 variable)
initial  value 4243.939271
iter  10 value 2656.830413
final  value 2656.829742
converged
# weights:  6 (2 variable)
initial  value 4714.145331
iter  10 value 3241.141811
final  value 3241.141714
converged
# weights:  6 (2 variable)
initial  value 8959.183214
iter  10 value 5947.212685
final  value 5947.195630
converged
```

```r
resultados %>%
  xtable::xtable(digits = c(0,rep(3,ncol(resultados))),
                 caption="Estimativas pelo método NPL",
                 label="tab:5") %>%
```

```
  print(include.rownames=F,
        comment=FALSE)
```

Tabela 7: Estimativas pelo método NPL

Grupo	LL	RC	RC_SE	T1	T1_SE
Grupo 1, 2, 3 e 4	-132.370	12.016	1.948	4.731	1.289
Grupo 1, 2, 3 e 4	-163.737	9.654	1.192	2.808	0.616
Grupo 1, 2, 3 e 4	-300.819	9.409	0.792	3.162	0.502

Para essa estimativa foi possível utilizar o fator de desconto correto $\beta = 0.9999$. Pelos resultados observados na Tabela 7, verifica-se que os valores são muito próximos daqueles observados na Tabela IX do trabalho de Rust, todavia com algumas diferenças. Além disso, o tempo de computação dos resultados se reduz sobremaneira, indicando a melhoria nesse quesito em relação ao método anterior.

v) Faça um gráfico de como as estimativas Hotz-Miller mudam a medida que você varia T e S. Para quais níveis de T e S suas estimativas ficam estáveis? Por que novos aumentos no valor de T não mudam suas estimativas?

3 Anexos

Abaixo estão todas as funções utilizadas na lista

```
# Definir a função de custo míope
# Fonte:
# https://github.com/waynejtaylor/Single-Agent-Dynamic-Choice/
# blob/master/Rust%20Data%20Generating%20Process.R
custo_miope=function(S, FCM, params, p){

  "A função custo míope computa o custo esperado associado com
  cada decisão para cada estado, retornando um conjunto de
  decisões/estados de custo.

   Inputs:
    * Um inteiro S, descrevendo os possíveis estados do ônibus.
    Na pratica, é o número de meses analisados
    * A função de custo de manutenção FMC, que tem como input
    um vetor de parâmetros e estados
    * O vetor params, como entrada para FMC. O primeiro elemento do vetor
      é o custo de substituição cs.
    * Um vetor (3x1) descrevendo os estados de transição de probabilidades
```

```
    Outputs:
    * Uma matriz (Sx2) contendo os custos de manutenção e troca para os N
      possíveis estados dos ônibus"

  # Definir o custo de substituição a partir dos
  # Parâmetros fornecidos
  cs = params[1]
  # Definir um vetor para receber os custos
  # de manutenção
  maint_cost = rep(NA,S)
  # Definir um vetor para receber os custos
  # de substituição
  repl_cost = rep(NA,S)

  for(s in 1:S){
    maint_cost[s] = FCM(s,params[-1])
    repl_cost[s] = cs
  }

  cbind(maint_cost,repl_cost)
}

# Possíveis formas para a função custo:
# Linear: FCM(s,theta)=theta_11*s
# Quadrática: FCM(s,theta)=theta_11*s+theta_12*s^2
# Exponencial: FCM(s,theta)=exp(theta_11*s)
# Log: FCM(s,theta)=log(theta_11+theta_12*s)

lin_cost=function(s,params) s*params[1]*0.001
quad_cost=function(s,params) (s*params[1]*0.001)+s^2*params[2]
exp_cost=function(s,params) exp(s*params[1]*0.001)
log_cost=function(s,params) log(params[1]*0.001 + s*params[2])

# Definição das escolhas de probabilidade, como função do vetor
# de custos

escolha_prob=function(cost_array){
```

```
  # Retorna a probabilidade de cada escolha,
  # condicional a um vetor de custos.

  S = nrow(cost_array)

  #toma a diferença pois 1) resultados são iguais

  cost = cost_array-apply(cost_array,1,min)

  # 2) mais estável com exp()
  util = exp(-cost)
  pescolha = util/rowSums(util)

  pescolha
}

# Função Contraction map
contraction_mapping=function(S,
                             p,
                             FCM,
                             params,
                             beta=beta,
                             threshold=1e-9,
                             suppr_output=FALSE){

  "Computa o valor esperado não-míope do agente para cada decisão possível
  e cada possível estado do ônibus.
  A iteração é realizada até a diferença obtida entre o passo
  anterior e o atual
  é menor que o limiar.
  Inputs:
  * Um número finito de estados S
  * Um vetor com as probabilidades de transição de estado
  p = [p(0), p(1), p(2), ..., p(k)] de tamanho k < S
  * Uma função de manutenção de custos FMC
  * Um vetor de parâmetros para a função custo
  * Um fator de desconto beta (opcional)
  * Um limiar para convergência (opcional)
```

```
Outputs:
* A convergência para a escolha de probabilidade forward-looking e
míope para cada estado, condicional aos 'params'"

achieved = TRUE

# Inicialização das matrizes de estado de transição
# ST_mat: descreve as probabilidades de transição de
# estado se os custos de manutenção ocorrem
# RT_mat: Estado regenerado para 0 se o custo de troca acontece.
# [a,b] = transição do estado "a" para "b"
ST_mat = matrix(0,S,S)
lp = length(p)
for(i in 1:S){
  for(j in 1:lp){
    if((i+j-1)<S)  ST_mat[i,i+j-1] = p[j]
    #sobre as colunas (probabilidades colapsadas)
    if((i+j-1)==S) ST_mat[i,i+j-1] = sum(p[j:lp])
  }
}

R_mat = cbind(1,matrix(0,S,S-1))

# Inicialização do valor esperado
# (que também é a decisão de custo do agente míope).
# Aqui, o componente forward-looking é inicializado em 0.
k = 0
EV = matrix(0,S,2)
EV_miope = EV_nova = custo_miope(S, FCM, params, p)

# Contraction mapping loop
while(max(abs(EV_nova-EV)) > threshold){
  # Guardar o valor esperado anterior
  EV = EV_nova
  # Obter a probabilidade de manutenção e
  # troca a partir do valor esperado anterior
  pescolha = escolha_prob(EV)
  # Computar o custo esperado para cada estado: vetor Nx1
```

```
    ecost = rowSums(pescolha*EV)
    # Computar os dois componentes da utilidade forward-looking:
    # No caso de manutenção, a utilidade dos futuros
    # estados ponderada pela
    # probabilidade de transição. No caso de troca,
    # a utilidade futura é a utilidade do estado 0
    futil_maint = ST_mat%*%ecost
    futil_repl = R_mat%*%ecost
    futil = cbind(futil_maint,futil_repl)
    # A utilidade futura é descontada por beta, e
    # adicionado ao custo míope.
    EV_nova = EV_miope + beta*futil
    k = k+1
    if(k == 10000) achieved = FALSE
  }

  if(!suppr_output){
    if(achieved){
      cat("Convergência atingida em ",k," iterações")
    } else {
      cat("CM não convergiu! Diferênça média = ",
          round(mean(EV_nova-EV),2))
    }
  }

  list(CP_forward=escolha_prob(EV_nova),CP_miope=escolha_prob(EV_miope))
}



# logit dinâmico
dynamiclogit=function(params,data,S,p,FCM){

  "Avalia os parâmetros de custos subjacente ao padrão de troca do motor
  do ônibus por um agente forward-looking.

  Inputs:
  * Base de dados: um 'data.frame' com:
```

```
  -escolha: nome da coluna contendo uma variável
  dummy de escolha endógena
  -estado:nome da coluna contendo uma variável de estado exógeno

* p: o vetor de transição de estado da variável exógena.
    Exemplo: p = [0, 0.6, 0.4] significa que o ônibus irá
    mudar para o próximo estado de milhagem com probabilidade 0.6,
    e para o segundo estado de milhagem 0.4.

* FCM: Função de custo de manutenção, que deve deve ter como
primeiro argumento o estado s e como segundo argumento um vetor
de parâmetros."

endog = data$troca
exog = data$estado

N=length(endog)
S=90

# Matrizes para acelerar a computação do log-likelihood

# Uma matriz (SxN) indicando o estado de cada computação
state_mat=matrix(0,S,N)

for(s in 0:(S-1)) state_mat[s+1,]=(exog==s)*1

# Uma matriz (2xN) indicanto com uma dummy a decisão
# tomada por cada agente (manutenção ou troca)
dec_mat = rbind(t(1-endog),endog)

"
A log-likelihood do modelo dinâmico é estimada em alguns passos:
1) Os parâmetros atuais entram na função  contraction mapping
2) A função retorna uma matriz de decisão
   de probabilidade para cada estado
3) Essa matriz é utilizada para computar a log-likelihood
das observações
4) A log-likelihood é somada entre os indivíduos
```

```
  "

  util = contraction_mapping(S=S,
                             p=p,
                             FCM=FCM,
                             params=params,
                             beta = beta,
                             suppr_output = TRUE)
  pescolha = util$CP_forward
  logprob = log(t(state_mat)%*%pescolha)
  -sum(logprob*t(dec_mat))
}



clogit=function(ydum,x,restx,namesb,sup_out=FALSE){

  "Estimativa do Logit condicional de McFadden's Conditional
  Alguns parâmetros podem ser restringidos com
  o algorítmo de otimização: Newton's com
  Gradiente e Hessiano analítico

  Fonte: https://github.com/waynejtaylor/Single-Agent-Dynamic-Choice/
  blob/master/clogit.R

  Input       ydum    - (nobs x 1) vetor da variável dependente categórica:
                        {1, 2, ..., nalt}
              x       - (nobs x (k * nalt)) matriz das variáveis
              explicativas irrestritas, seguindo as alternativas de ydum.
              restx   - (nobs x nalt) vetor da soma das variáveis explicativas
              com restrição de parâmetro igual a 1
              namesb  - (k x 1) vetor com o nome dos parâmetros
              sup_out - TRUE/FALSE mostrar ou não o output
  Output      best    - (k x 1) vetor com as estimativas ML
              varest  - (k x k) matrizes de variância e covariância"

  # Tolerância
  cconvb = 1e-6
```

```
# Toerância do zero
myzero = 1e-16
# Número de observações
nobs = length(ydum)
# Número de alternativas
nalt = max(ydum)
# Número de parâmetros
npar = ncol(x)/nalt
# Teste para parâmetros
if(npar!=length(namesb))
  cat("Erro: Número e nome dos parâmetros não batem ")

# Objeto para receber Valorda soma
xysum = rep(0,npar)
# Passo 1
j=1
# Iniciar loop
while(j<=nalt){
  # Somar
  xysum = xysum + colSums((ydum==j)*x[,(npar*(j-1)+1):(npar*j)])
  j=j+1
}

# Iteração
iter=1
# Número de iterações
criter = 1000
# Likelihood
llike = -nobs
# Matriz para os parâmetros
b0 = matrix(0,npar,1)

# Iniciar looping
while(criter>cconvb){

  # A cada passo, mostrar
  if(!sup_out){
    cat("Iteração               = ",iter,fill=TRUE)
```

```
  cat("Função Log-Likelihood  = ",llike,fill=TRUE)
  cat("Norm of b(k)-b(k-1)      = ",criter,fill=TRUE)
}

#Computando as probabilidades
phat = matrix(0,nobs,nalt)
j=1
# Iniciar looping
while(j<=nalt){
  phat[,j] = x[,(npar*(j-1)+1):(npar*j)]%*%b0 + restx[,j]
  j=j+1
}

# Valores estimados
phat = phat - apply(phat,1,max)
phat = exp(phat)/rowSums(exp(phat))

# Computar a média
sumpx = matrix(0,nobs,npar)
xxm = matrix(0,npar,npar)
llike = 0
j=1
# Loop
while(j<=nalt){
  xbuff = x[,(npar*(j-1)+1):(npar*j)]
  sumpx = sumpx + phat[,j]*xbuff
  xxm = xxm + t(phat[,j]*xbuff)%*%xbuff
  llike = llike+sum((ydum==j)*
                    log((phat[,j]>myzero)*phat[,j]
                        +(phat[,j]<myzero)*myzero))
  j=j+1
}

# Computar o gradiente
d1llike = xysum - colSums(sumpx)

# Computar a hessiana
d2llike = - (xxm - t(sumpx)%*%sumpx)
```

```
    # Iteração Gauss
    b1 = b0 - solve(d2llike)%*%d1llike
    criter = sqrt(t(b1-b0)%*%(b1-b0))
    b0 = b1
    iter = iter + 1
  }

  # Erro-padrão
  Avarb  = solve(-d2llike)
  sdb    = sqrt(diag(Avarb))
  tstat  = b0/sdb

  if(!sup_out){
    numyj  = as.vector(table(ydum))
    logL0  = sum(numyj*log(numyj/nobs))
    lrindex = 1 - llike/logL0

    cat("----------------",fill=TRUE)
    cat("Nº de Iterações     = ",iter,fill=TRUE)
    cat("Nº de observações   = ",nobs,fill=TRUE)
    cat("Função Log-Likelihood  = ",llike,fill=TRUE)
    cat("Likelihood Ratio Index   = ",lrindex,fill=TRUE)
    cat("----------------",fill=TRUE)
    print(data.frame(Parameter=namesb,
                     Estimate=round(as.numeric(b0),3),
                     StandardErrors=round(sdb,3),
                     tratios=round(as.numeric(tstat),3)))
    cat("----------------",fill=TRUE)
  }

  list(b0=b0,Avarb=Avarb,L=llike)
}


library(pracma)

npl_sing=function(inda,indx,zmat,pini,bdisc,fmat,names){
```

```
"
Estimativa dos parâmetros estruturais por Máxima Verossimilhança
de escolha discreta de programação dinâmica (single-agent)
usando o algorítmo NPL (Aguirregabiria and Mira, Econometrica, 2002)

Fonte: https://github.com/waynejtaylor/Single-Agent-Dynamic-Choice/
blob/master/npl_sing.R

------------------------------------------------------------------

INPUTS:
  inda    - (nobs x 1) Vetor com o índice das escolhas
  discretas (1,...,J)

  indx    - (nobs x 1) Vetor com o índice da variável
  de estado (1,..,S)

  zmat    - (zmat1,zmat2,...,zmatJ) matrizes com os
  valores das variáives
            z(a=j,x)
          obs.: cada zmat possui J colunas
          para representar a utilizade da escolha j
          dada uma ação a

  pini    - (numx x J) vetor com as
  estimativas iniciais de probabilidade Pr(a=j|x)

  bdisc   - Fator de desconto (entre 0 e 1)

  fmat    - (fmat1,fmat2,...,fmatJ) matriz com
  a probabilidade condicional de transição

  names   - (npar x 1) vetor com o nome dos parâmtros

 OUTPUTS:
  Lista de tamanho K na qual a k'th estrada contém:

  tetaest - (npar x 1) matrizes com os parâmetros estruturais
```

```
  do k'th passo

  varest  - (npar x npar) matrizes de variância e covariância
  do k'th passo

  pest    - (numx x J) matriz com as probabilidades
  Pr(d=1|x),...,Pr(d=J|x) estimadas k'th passo
-------------------------------------------------------------"

# Número de parâmetros
npar = length(names)
# Número de observações
nobs = length(inda)
# Número de escolhas
nchoice = max(inda)
# Teste para a matriz zmat
if(ncol(zmat)!=(npar*nchoice)){
  print("Erro: O número de colunas em 'zmat' não bate",fill=TRUE)
  print("com o número de: 'escolhas * nº parâmetros'",fill=TRUE)
}

# Tolerância para o valor zero
myzero = 1e-9
# Constante de Euler
eulerc = 0.5772
# Tamanho do pini
numx = nrow(pini)
# Número de interações
convcrit = 1000
# Tolerância convergência
convcons = 1e-9
# Theta estimado
tetaest0 = matrix(0,npar,1)
# Objeto do output
out = NULL

#-----------------------------------------------------
#             Estimativa dos parâmetros estruturais
```

```
#-----------------------------------------------------
# COntrole inicial
ks=1

# Iniciar loop

while(convcrit>=convcons){

  cat("--------------------",fill=TRUE)
  cat("Estimador: ESTÁGIO =",ks,fill=TRUE)
  cat("--------------------",fill=TRUE)

  #1. Obter as matrizes
  #"A=(I-beta*Fu)", "Bz=sumj{Pj*Zj}" e o vetor Be=sumj{Pj*ej}
  #-----------------------------------------------------

  i_fu = matrix(0,numx,numx)
  sumpz = matrix(0,numx,npar)
  sumpe = matrix(0,numx,1)
  j=1

  # Loop da escolha

  while (j<=nchoice){
    #coluna de referência
    i_fu = i_fu + pini[,j]*fmat[,(numx*(j-1)+1):(numx*j)]
    sumpz = sumpz + pini[,j]*zmat[,(npar*(j-1)+1):(npar*j)]
    # Tratamento para o log
    sumpe = sumpe + pini[,j]*(eulerc - log(pini[,j]+myzero))
    j=j+1 ;
  }

  i_fu = diag(numx) - bdisc * i_fu

  #2. Resolução dos sistemas
  #"A*Wz = Bz" e "A*We = Be" usando  a decomposição CROUT
  #---------------------------------
```

```
i_fu = lu(i_fu)
wz = solve(i_fu$L,cbind(sumpz,sumpe))
wz = solve(i_fu$U,wz)

we = wz[,npar+1]
wz = wz[,1:npar]

#OU:
# we=solve(i_fu,sumpe)
# wz=solve(i_fu,sumpz)

#3. Computando
#"ztilda(a,x) = z(a,x) + beta * F(a,x)'*Wz"
# and "etilda(a,x) = beta * F(a,x)'*We"
#---------------------------------------

ztilda = matrix(0,numx,nchoice*npar)
etilda = matrix(0,numx,nchoice)
j=1
while(j<=nchoice){
  ztilda[,(npar*(j-1)+1):(npar*j)] =
    zmat[,(npar*(j-1)+1):(npar*j)]+
    bdisc*fmat[,(numx*(j-1)+1):(numx*j)]%*%wz
  etilda[,j] = bdisc * fmat[,(numx*(j-1)+1):(numx*j)]%*%we
  j=j+1
}

#4. Observações de "ztilda" e "etilda"
#---------------------------------------

zobs = ztilda[indx,]
eobs = etilda[indx,]

#---------------------------------------
#5. Estimação de Pseudo Maximum Likelihood

clogitout=clogit(inda,zobs,eobs,names)
tetaest1=clogitout$b0
```

```
    varest=clogitout$Avarb
    L=clogitout$L

    #6. Recomputando as probabilidades
    #---------------------------------------

    pini = matrix(0,numx,nchoice)
    j=1
    while(j<=nchoice){
      pini[,j] = ztilda[,(npar*(j-1)+1):(npar*j)]%*%tetaest1 + etilda[,j]
      j=j+1
    }
    pini = pini - apply(pini,1,max)
    pini = exp(pini)
    pini = pini/rowSums(pini)

    #7. Critério de convergência
    #---------------------------------------
    convcrit = max(abs(tetaest1-tetaest0))
    tetaest0 = tetaest1
    cat("Critério NPL =",convcrit,fill=TRUE)

    #8. Salvar o output do estágio k
    #---------------------------------------
    out[[ks]]=list(tetaest=tetaest1,varest=varest,pini=pini,L=L)

    # Seguir para o próximo passo
    ks=ks+1
  }

  # Retornar o output
  out
}
```