Avaliação de Decisões Estratégicas sob Incerteza: Análise das Contribuições da Modelagem Exploratória (EMA) e Robust Decision Making (RDM)

Pedro Nascimento de Lima

Versão Preliminar - R

Abstract

Este documento contém os resultados da dissertação, e é escrito diretamente no R. Os capítulos do projeto da dissertação (1,2 e 3) não estão reproduzidos por completo neste documento, mas sua estrutura e argumentos utilizados está representado para alinharmos o trabalho como um todo, e decidirmos o local de cada argumento. Por hora, vou gerar e escrever o documento no R para agilizar a geração dos gráficos e fórmulas. Após a estabilização do documento, vou levá-lo para o word e formatar.

Contents

1 Introdução:

Problematização sobre os desafios que a incerteza impõe à tomada de decisão estratégica.

1.1 Objeto e Questão de Pesquisa:

- Objeto: Avaliação de Decisões Estratégicas sob incerteza profunda. Uso o framework de processo de decisão estratégica do mintzberg para localizar o objeto da pesquisa.
- Questão de Pesquisa: "Quais são as contribuições da Modelagem Exploratória (EMA) e do Robust Decision Making (RDM) para a avaliação de decisões estratégicas organizacionais em situações de incerteza profunda?"

1.2 Objetivos

1.2.1 Objetivo Geral

"Analisar as contribuições da EMA e do RDM para a avaliação das decisões estratégicas em situações de incerteza profunda."

1.2.2 Objetivos Específicos

a) identificar abordagens para avaliação de decisão estratégica sob incerteza profunda;
b) instanciar o RDM no contexto empresarial;
c) avaliar a instanciação do RDM no contexto empresarial;
d) identificar heurísticas contingenciais na aplicação do RDM no ambiente empresarial.

1.3 Justificativa

Argumentação sobre as limitações das abordagens para tomada de decisão de incerteza. Linha Geral de Argumentação:

- Abordagens Atuais apresentam limitações sob incerteza profunda;
- Existe o RDM (e outros métodos);
- Não existe menção ao RDM na literatura de estratégia em negócio;
- O trabalho contribui realizando uma "exaptação" da abordagem.

2 Fundamentação Teórica

2.1 Avaliação de Decisões Estratégicas Sob Incerteza Profunda

2.1.1 Avaliação de Decisões Estratégicas

2.1.2 Níveis de Incerteza e Incerteza Profunda

2.2 Abordagens para Avaliação de Decisão sob Incerteza Profunda

2.2.1 Identificação de Artefatos

2.2.2 Contextos de Aplicação do RDM

2.3 RDM - Robust Decision Making (. . .)

3 Método de Pesquisa (. . .)

4 Contexto de Aplicação - Indústria da Manufatura Aditiva

Links Interessantes para ler e citar:

5 a complex table with merged cells, column/row separators, and custom alignments and widths

Impressão 3D na manufatura: https://www.technologyreview.com/s/604088/the-3-d-printer-that-could-finally-change-manufac

- Evolução Tecnológica das Impressoras 3D;
- Número de Patentes relacionadas à impressão 3D: (Gráfico).
- Investimento em P & D de empresas de impressoras 3D;
- Investimento e P e D gera patentes, que expiram com o tempo:

https://3dprintingindustry.com/news/more-3d-printing-patents-are-expiring-soon-heres-a-roundup-96561/

https://techcrunch.com/2016/05/15/how-expiring-patents-are-ushering-in-the-next-generation-of-3d-printing/

https://futurism.com/expiring-patents-set-to-improve-3d-world/

https://qz.com/106483/3d-printing-will-explode-in-2014-thanks-to-the-expiration-of-key-patents/

https://www.finnegan.com/en/insights/how-patents-die-expiring-3d-printing-patents.html

Patentes na Impressão 3D: Este artigo mostra os principais players da impressão 3D. https://www.finnegan.com/en/insights/3d-printing-patent-landscape.html

História da Formlabs: https://techcrunch.com/2012/09/27/3d-printer-form-1-gets-6x-its-100k-funding-goal-on-kickstarter-in-c

Alguns argumentam que o número de patentes expirando pode influenciar a queda dos preços e gerar interesse pela impressão 3D.

Um exemplo é a impressoã de materiais metálicos. A tecnologia foi criada inicialmente em 1984,

- Nova Impressora 3D que imprime 10 x mais rápido que as demais: http://www.sciencedirect.com/science/article/pii/S2214860416303220

https://www.technologyreview.com/the-download/609607/blink-and-youll-miss-how-fast-this-souped-up-3-d-printer-makes-pr?utm_source=facebook.com&utm_medium=social&utm_content=2017-11-30&utm_campaign=Technology+Review

- Difusão da Impressão 3D em novos mercados e nichos;
- Gráfico dos Nichos de Mercado da Impressão 3D e demonstração de que eles estão em fases diferentes;
- Melhoria dos Processos propiciado pela impressão 3D;
- Bico de injeção da GE com tecnologia superior;

As discussões desta seção provavelmente irão para o final do capítulo 2.

5.1 Comportamento da Demanda de Impressora 3D

5.2 Principais Players do Mercado

5.3 Comportamento de Variáveis Relevantes

5.4 Sub-divisão dos mercados potenciais da Impressão 3D

5.5 Delimitações do Trabalho

5.6 Questões relevantes levantadas para a simulação.

Questões não respondidas que o meu trabalho pode responder:

- Como pode se comportar a demanda por impressoras 3D?
- Que Estratégia de Capacidade um Player deve adotar para este ramo: Estratégia Agressiva de penetração no mercado ou estratégia "Conservadora".
- Esperar o cenário melhor se configurar para agir ou agir para conquistar market share de modo preemptivo?
- Quais são as incertezas mais importantes para a determinação da estratégia de capacidade mais adequada?

As questões acima devem levar à escolha da simulação de dinâmica de sistemas como abordagem ideal. Não devem ser colocadas questões acima que a análise não irá ajudar a responder.

Quais players simular.

Que aspectos simular ou não

6 Revisão de Modelos

Em resposta às necessidades do item anterior, os modelos de difusão de novos produtos devem ser avaliados, culminando no modelo do Sterman (XXX). As características dos modelos podem ser brevemente descritas para ajudar nesta delimitação.

[Quadro de Comparação dos Modelos]

(Escrever no Word para facilitar as citações.)

Falar sobre cada modelo e mostrar o Quadro da análise dos modelos. Considerar que cada um dos modelos considera e suas contribuições e limitações para o trabalho atual. Ressaltar o que o Sterman considera e que os demais não consideram para justificar a escolha do Sterman como ponto de partida.

O modelo proposto inicialmente por Sterman (XX) foi utilizado como ponto de partida deste trabalho, por possuir uma série de características desejáveis para este trabalho. Em primeiro lugar, o modelo não é restrito a monopólios, como o modelo de Bass (XX) e outros modelos deste trabalho (identificar e citar aqui). Além disso, o modelo possui uma estrutura de dinâmica competitiva considerando a interação de diversos fatores presentes na Indústria da Manufatura Aditiva, incluindo curvas de aprendizagens, diferentes players expandindo sua capacidade produtiva em função da demanda prospectada no mercado.

Trabalho	Bass (1969)	Mahajan Muller (1996)	Dattée, Birdseye (2007)	Maier (1998) - Modelo de Competição	Maier (1998) - Modelo de Substituição	Cui, Zhao, Ravichandran (2011)	Sterman (2007)
Objeto original		Timing de Substituição de gerações de novos produtos com inovação tecnológica. (new product launch strategy)	Substituições Tecnológicas (technological substitutions)	Modelos de Difusão de Novos produtos (new product difgusion models).	Dinâmica de substituição de produtos novos por modelos antigos, assumindo que há monopólio de mercado.	Dynamic New Product Launch Strategies	
Principal Crítica aos demais modelos.		O modelo original de Bass não captura a sucessão de diferentes gerações de produtos.	Simplificam em demasia a heterogeneidade do mercado.	Não consideram a entrada de outros concorrentes no mercado.	Não consideram a entrada de novos modelos no mercado, e o tradeoff entre introduzir um produto cedo ou tarde.	Na maioria das vezes, não consideram estratégias dinâmicas.	
Modelos de Referência Citados.		Bass (1969), Wilson e Norton	Bass (1969) (modelo de difusão), Fischer e Pry (modelo de competição entre tecnologias).	Bass (1969), Milling (1986; 1987; 1989); Maier (1995) e Millinr e Maier (1996)	Fisher e Pry (1971), Norton e Bass (1987)		
X - Incertezas		Tamanho relativo dos mercados potenciais, margem do produto, parâmetros de difusão e substituição.	Heterogeneidade da população de possíveis clientes das substituições. Diferentes classes de clientes podem valorizar aspectos do produto de modo diferente, levando a dinâmicas de adoção diversas.	Tempo de Entrada de outros concorrentes para a divisão do mercado. Market share dos concorrentes em função de seu "coeficiente de inovação".	Tamanho potencial do mercado, Market Share, Multiplicador de Substituição, Tempo de obsolescência, Entrade de novos clientes potenciais, Capacidade Técnica dos Produtos e Preços.		
L - Estratégias / Decisões		Timing entre introdução de novos modelos de produtos com inovação tecnológica.	Obtenção de primeiros usuários que são formadores de opinião para amplificar o efeito da comunicação dentro de uma rede.	Estratégias de Precificação, organentação para pesquisa e desenvolvimento, tempo de entrada no mercado, e estratégias de divulgação.			
R - Relações		Mesmas relações contidas no modelo de bass, acrescentadas da relação de substituição de máquinas.	Relações entre fatores sociais (credibilidade, disponibilidade de informação) e a adoção de uma nova tecnologia. Adoção da tecnologia é moderada por um índice de performance da tecnologia e o seu custo.	Precificação, Esforços de Marketing e Delays na Entrega influenciam a probabilidade de compra. A competição (novos entrantes no mercado) também é considerada.	Relações entre incertezas adotadas, e vendas de novos modelos de produtos. O multiplicador de substituição é calculado em função da "capacidade técnica" do novo modelo e de seu preço.		
M - Métricas		Número Total de Produtos Vendidos, por geração.	Vendas totais por tecnologia, Número total de consumidores usuários.	Vendas, número de clientes.			

Figure 1: Temporario - Completar Quadro e Analisar Aqui

7 Ferramenta Computacional para a Análise RDM

O objetivo desta seção é descrever a ferramenta computacional desenvolvida no âmbito desta dissertação para viabilizar a operacionalização da análise RDM. A decisão por desenvolver a análise nesta dissertação por meio deste ambiente aberto, ainda que em princípio mais custosa, teve por objetivo realizar a análise RDM com a máxima independência possível, sem recorrer à ferramentas terceiras ou privadas. Além disto, o desenvolvimento desta ferramenta computacional permitirá que os resultados desta dissertação sejam reproduzidos. Recomenda-se ao leitor interessado que acesse a ferramenta disponível no link (http://bit.ly/pnldissert) Deste modo, procura-se atender aos requisitos de reprodutibilidade em trabalhos baseados em simulação computacional preconizados por Rahmandad e Sterman (2012).

A primeira barreira para a realização da Análise RDM é a disponibilidade de ferramentas computacionais amigáveis para a operacionalização da análise exploratória. Embora existam frameworks de desenvolvimento úteis para a modelagem exploratória (como o EmaWorkbench (KWAKKEL, 2013) o OpenMORDM (HADKA et al., 2015) e o Rhodium(XXX)), tais ferramentas implicam em empecilhos para a utilização no contexto deste trabalho. Em primeiro lugar, estas ferramentas requerem que seu usuário final programe o modelo computacional e insira os parâmetros diretamente no código fonte. Embora propiciem um ambiente de desenvolvimento adequado para programadores proeficientes nas suas respectivas linguages de programação, estas bibliotecas carecem de interfaces para que os usuários finais interajam com os inputs da simulação (ex.: alterem os parâmetros de entrada e estratégias a serem simuladas), e avaliem imediatamente o resultado das simulações.

A ferramenta EmaWorkbench, desenvolvida na linguagem python não possui interface gráfica, não suporta integração com o software de dinâmica de sistemas iThink, ou com modelos desenvolvidos na linguagem R. Neste sentido, a ferramenta requer que o modelo seja desenvolvido em uma ferramenta como o Vensim, Excel ou um modelo utilizando a linguagem Python.

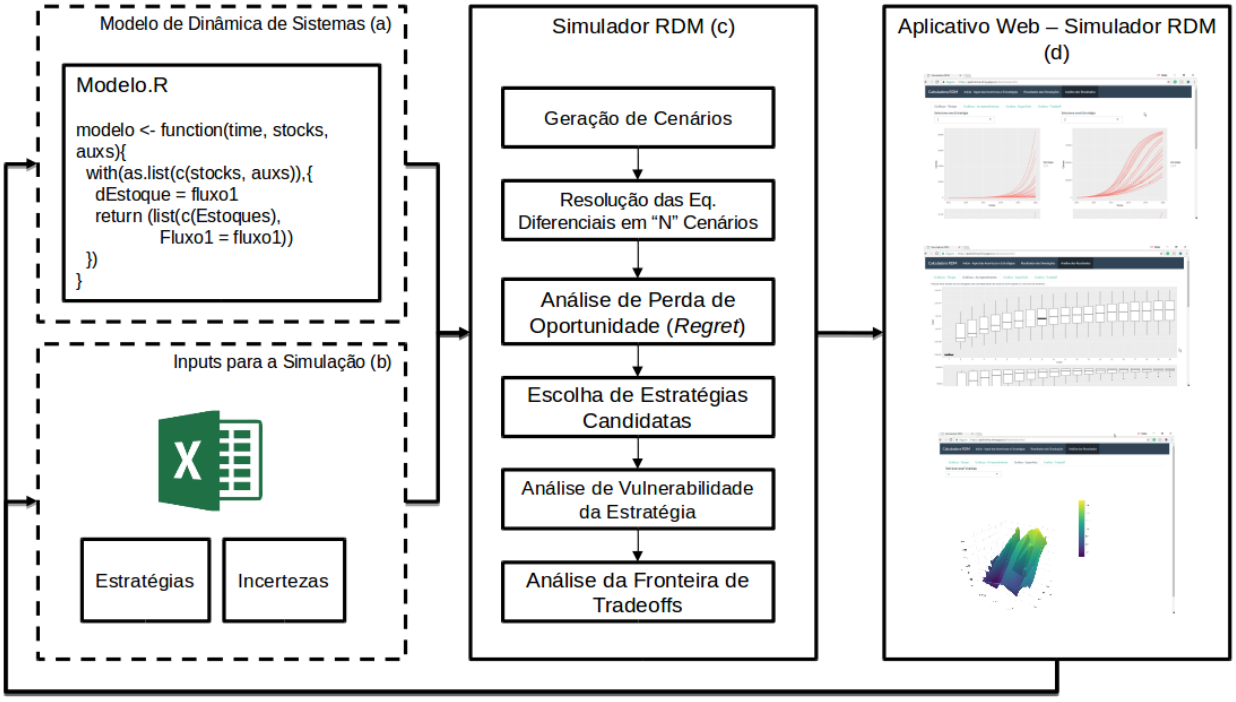


Figure 2: Projeto Modular da Ferramenta Computacional para a Análise RDM

Considerando a necessidade de flexibilidade durante a execução deste trabalho, o pesquisador optou por desenvolver rotinas computacionais próprias utilizando a linguagem R e bibliotecas de código aberto disponíveis no repositório CRAN. A linguagem R possui bibliotecas para a integração numérica do modelo computacional (biblioteca deSolve), para a calibração do modelo (FME), para a disponibilização dos resultados em um aplicativo web (shiny), e para a visualização interativa dos resultados (ggplot2, plotly). Utilizando tais bibliotecas em conjunto, foi possível implementar as rotinas computacionais para a operacionalização do RDM, cuja estrutura é ilustrada na Figura (XXX).

7.1 Módulos da Ferramenta Computacional

A ferramenta computacional foi projetada com o objetivo de receber uma planilha de inputs de dados (contendo a definição de estratégias a serem simuladas e incertezas a serem consideradas), e a partir do modelo computacional desenvolvido, rodar os passos da análise RDM com a maior grau de automação possível. A seguir são descritos os quatro principais módulos da ferramenta (ilustrados na Figura (XX)), e suas principais funções, com o propósito de viabilizar seu uso ou adaptação em trabalhos futuros.

O primeiro componente necessário para a análise RDM é um modelo de simulação computacional. Por parte do RDM (e da análise exploratória em geral), não há uma limitação ou especificação quanto ao tipo de modelo a utilizar. Conforme Lempert (XXX 2006) esclarece, o framework de análise RDM pressupõe que modelos de simulação de "complexidade arbitrária" podem ser utilizados pela análise, desde que sejam capazes de relacionar decisões da empresa à métricas de resultado. A ferramenta computacional em questão propõe-se a suportar especificamente a utilização de modelos de dinâmica de sistemas desenvolvidos na linguagem R, de modo compatível à biblioteca de integração numérica deSolve.

O segundo componente (b) trata-se de uma planilha com formato padronizado, contendo as estratégias a serem simuladas e incertezas, incluindo valores máximos e mínimos para cada parâmetro. Esta planilha possui duas entradas de dados, com o propósito de permitir a entrada de incertezas (elemento X do framework

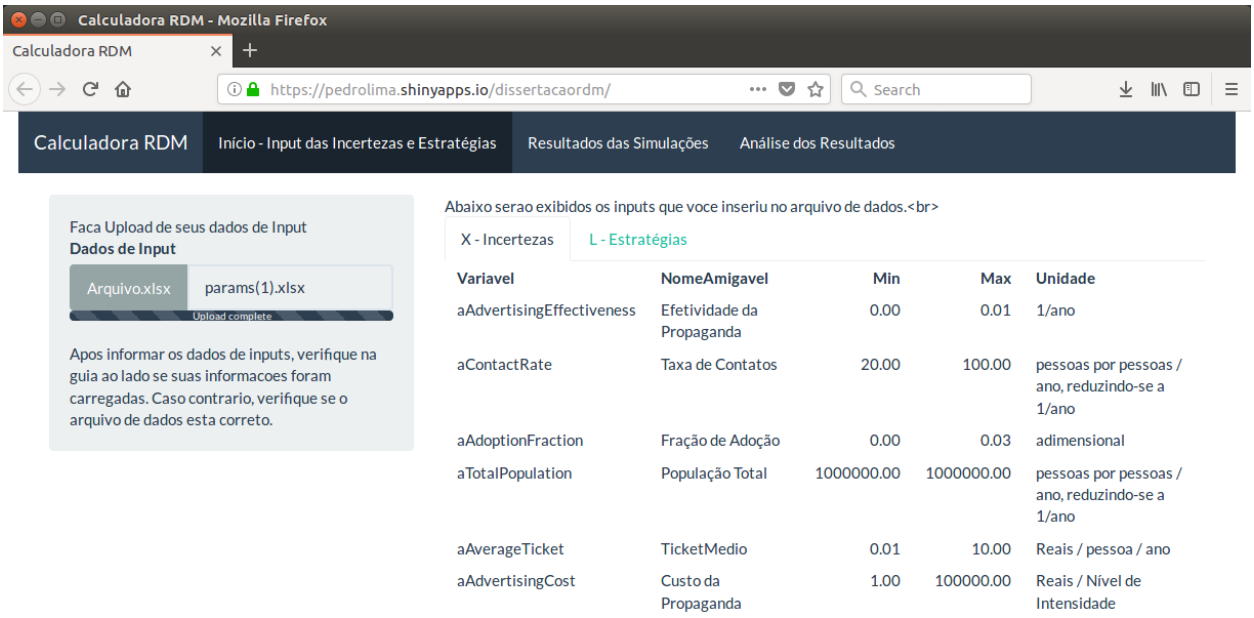


Figure 3: Tela de Inputs da Ferramenta Computacional

XLRM) e de estratégias (elemtno L do framework XLRM). A tabela de incertezas deve conter uma linha por variável considerada incerta, e seus ranges plausíveis, como é ilustrado no Quadro (XX).

Table 1: Entrada de Incertezas (X)

Variavel	**Nome Amigável**	**Min**	**Max**	**Unidade**
Incerteza1	Incerteza ABC	5	10	R$
Incerteza2	Incerteza XYZ	20	30	venda / pessoa
...	...	...	...	...
Incertezan	Incerteza xyz	0	1	% Market Share
Parametrofixo	Parametro ABC	2,5	2,5	Número de Pessoas

Uma vez enviado à ferramenta computacional esta tabela é exibida na ferramenta web, como pode ser observado na Figura (XX).

A estruturação deste input neste formato permite que um número arbitrário de parâmetros incertos seja utilizado pelo modelo, e que o usuário possa alterar os parâmetros minimos e máximos e observar o impacto desta alteração em relação à análise realizada sem a necessidade de alterar o código fonte do modelo. O segundo elemento da entrada de dados consiste na tabela de estratégias a simular, que é ilustrada no Quadro (XX).

Table 2: Entrada de Estratégias (L)

Lever	**LeverCode**	**Variavel1**	**...**	**Variaveln**
1	Estratégia 1	1	0	0
2	Estratégia 2	1	0	1
3	Estratégia 3	0	1,5	1,5
...	...	...	...	...
n	Estratégia n	0	2,5	3

Neste segundo quadro, cada linha da tabela representa uma estratégia, ou seja, uma combinação única de decisões a serem simuladas em cada um dos "n" futuros plausíveis definidos. As colunas "Lever" e "LeverCode" são fixas identificam a estratégia a ser simulada. As demais colunas correspondem à nomes de variáveis, que devem corresponder aos nomes constantes no modelo computacional, e os valores que estas variáveis assumirão.

A partir do modelo computacional (presente na calculadora web) e dos inputs informados, a calculadora executa uma série de análises para a execução da análise RDM. O quadro (XX) sintetiza o papel de cada uma destas etapas. A seção de análise dos resultados neste trabalho detalhará o significado de cada uma destas etapas.

Etapa	**Função da Etapa**
Geração de Casos	Nesta etapa a técnica Latin Hypercube Sampling (Citar XX) é usada para gerar um conjunto de casos contra os quais cada estratégia será testada. Todas as incertezas informadas na planilha são variadas simultâneamente de modo a representar uma ampla gama de situações às quais as decisões da empresa poderão ser submetidas.
Resolução das Equações Diferenciais	Para cada um dos casos gerados, o algoritmo emprega a biblioteca deSolve para a integração numérica do conjunto de equações indicados no modelo. Nesta etapa, a variável de interesse é calculada (ex.: Valor Presente Líquido)
Análise de Perda de Oportunidade	Nesta etapa o algoritmo calcula a perda de oportunidade (regret) de cada estratégia em cada cenário. Desta maneira, estima-se o valor monetário perdido pela empresa por não escolher a melhor estratégia dentre as disponíveis para o cenário em questão.
Escolha de Estartégias Candidatas	A partir da perda de oportunidade calculada, uma estratégia candidata é selecionada dentre as disponíveis, utilizando-se um critério (o critério adotado por lempert (menor percentil 75%) é adotado por padrão).
Análise de Vulnerabilidade da Estratégia	Este processo emprega o algoritmo PRIM para identificar cenários que melhor caracterizam as condições nas quais a estratégia candidata tem performance ruim. Este processo não pode ser automatizado completamente, devido à característica iterativa do algoritmo PRIM.
Análise da Fronteira de Tradeoffs	Considerando a caracterização da vulnerabilidade da estratégia escolhida, a fronteira de tradeoffs é calculada exibindo as estratégias que levam à uma menor perda de oportunidade no cenário onde a estratégia candidata é ruim.

Destaca-se que os componentes (a) e (b) podem ser modificados conforme o caso a ser analisado, sem a necessidade de reprogramar todas as funções do Simulador (c), nem do aplicativo web desenvolvido (d). Esta seção não detalhará cada um dos componentes e análises propiciadas pela ferramenta computacional, as quais serão evidenciadas nas seções de análise seguintes.

[Parágrafo sobre a análise de perda de oportunidade para a definição da estratégia mais robusta segundo um

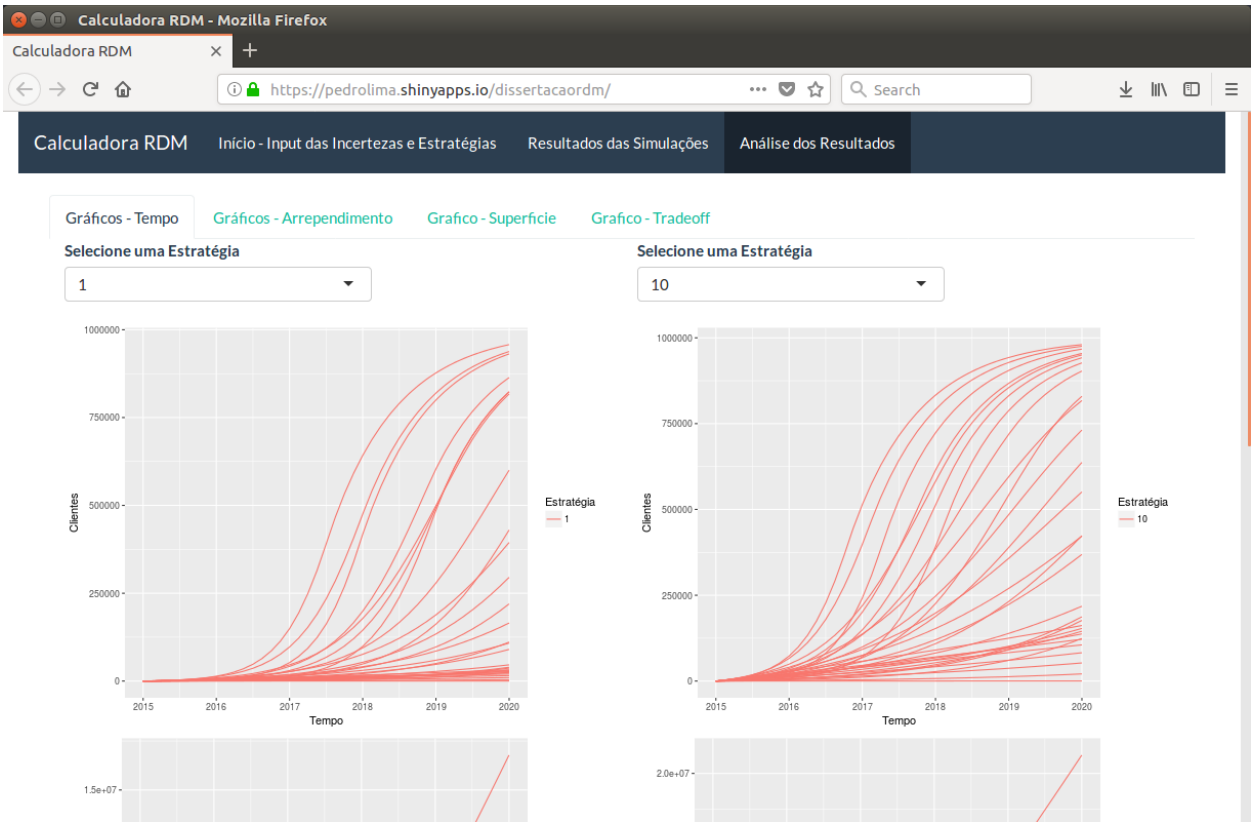


Figure 4: Comparação de Estratégias em "N" cenários utilizando a Ferramenta Computacional

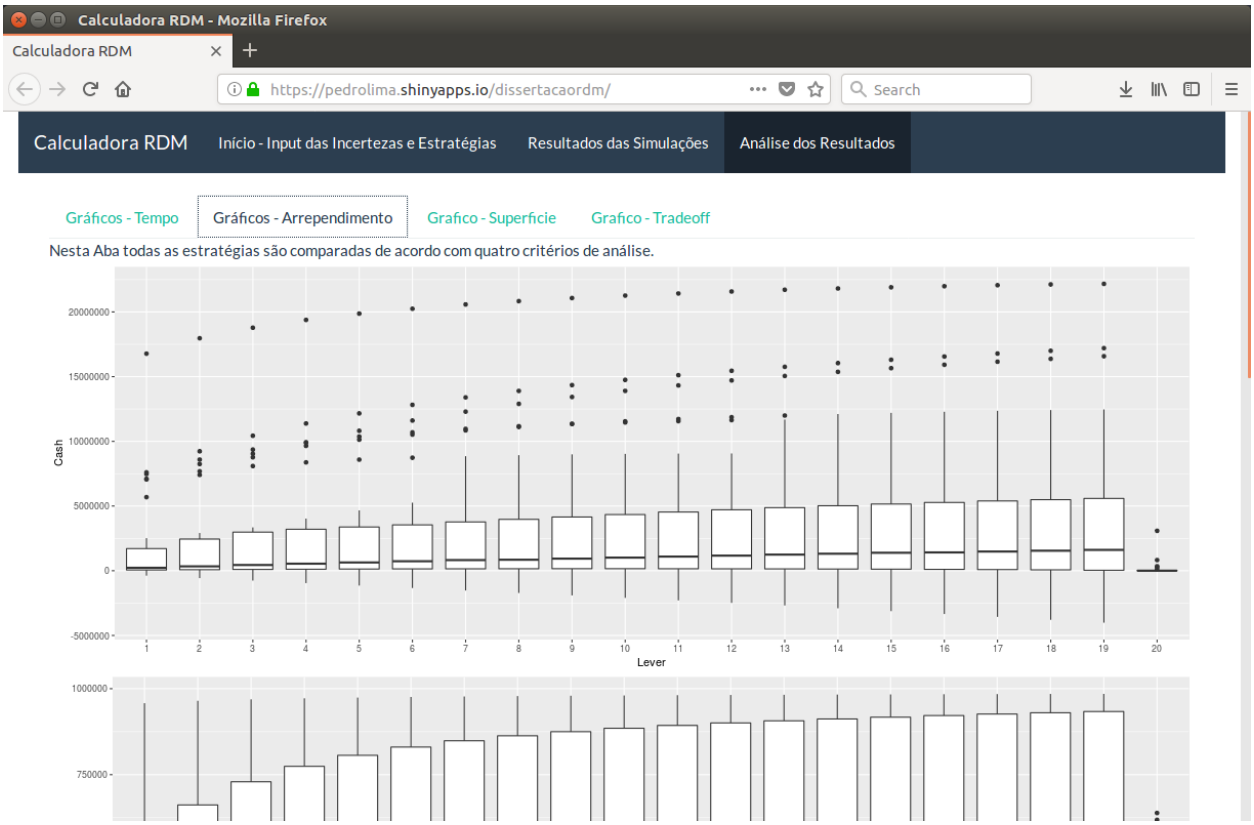


Figure 5: Análise de Perda de Oportunidade das Estratégias

determinado critério.]

8 Modelo da Competição na Indústria de Impressoras 3D

Esta seção descreve o modelo computacional empregado neste trabalho. Como f na seção 2.X.X, este trabalho utilizou como ponto de partida o modelo proposto por Sterman (XX), visto que este possui uma série de características aplicáveis à indústria da manufatura aditiva. Inicialmente, a estrutura geral do modelo é delineada, e o papel e funcionamento de cada um de seus módulos é sintetizado. Em seguida, a formulação matemática do modelo é justificada, e as modificações realizadas em relação ao modelo original são explicitadas.

8.1 Diagrama de Fronteiras do Modelo

A Figura (XX) ilustra os módulos do modelo e suas principais relações. Esta seção introduzirá as principais característstsicas do modelo, e argumentará sua relação com a indústria da manufatura aditiva. Além disto, a seção definirá as principais relações existentes entre os módulos e justificará a decisão pela inclusão de cada um destes módulos no modelo. Em seguida, a formulação matemática de cada um dos módulos será detalhada. Finalmente, esta seção também sintetizará as modificações empregadas no modelo original de Sterman (xx), justificando tais alterações.

Uma primeira característica importante para a compreensão do modelo é a escolha pela desagregação da maioria de seus módulos em diferentes players produtores de impressoras 3D. Tal desagregação permite

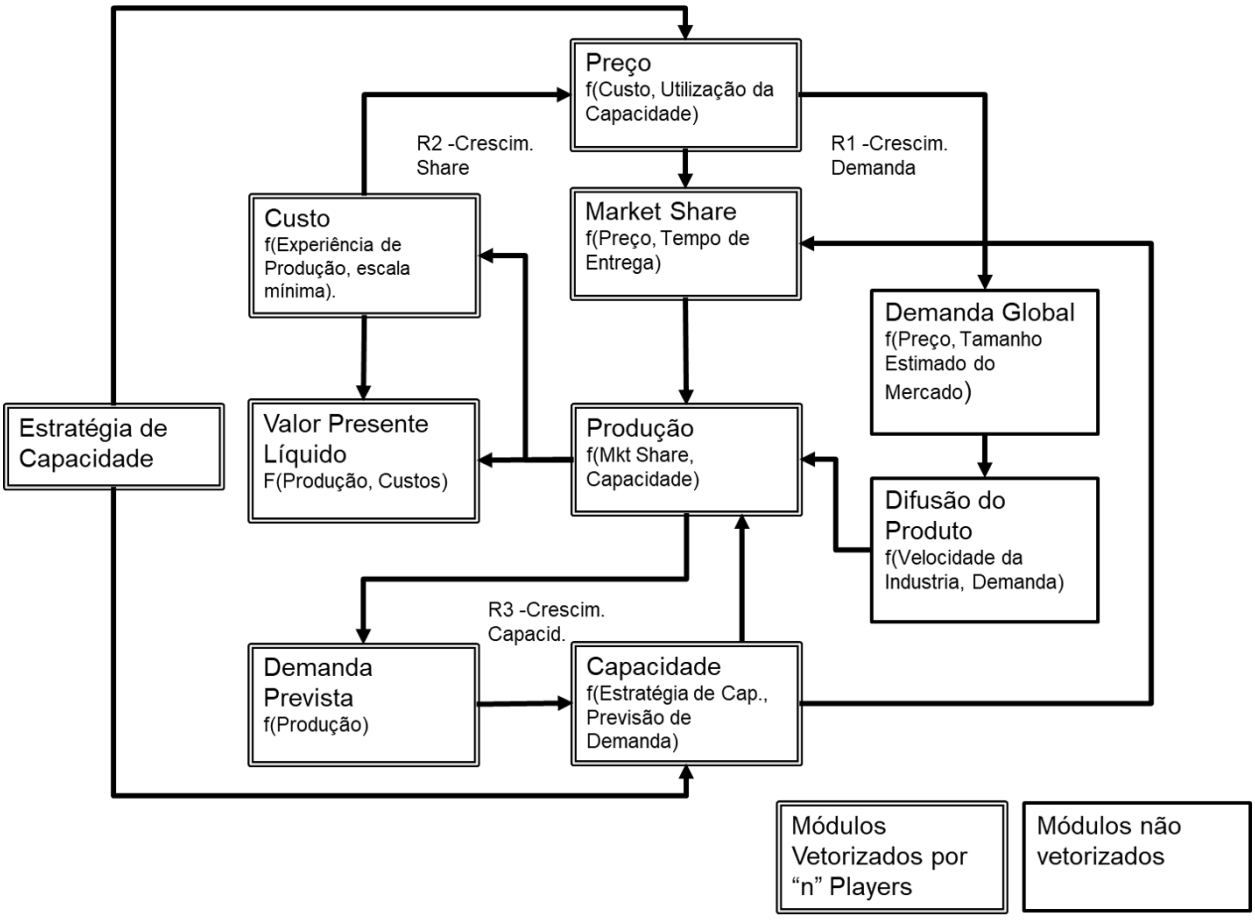


Figure 6: Modelo de Dinâmica Competitiva - Diagrama de Fronteiras

que o modelo simule a performance individual de players, e não apenas o comportamento agregado da indústria. Desta maneira, o modelo permite simular a interação entre decisões estratégicas dos diversos players simultâneamente.

Um segundo aspecto importante para a compreensão do modelo é que o mesmo ocupa-se de decisões estratégicas relacionadas à capacidade produtiva da empresa. Em específico, o modelo ocupa-se de analisar estratégias de crescimento de capacidade agressivas versus estratégias conservadoras. Adotando uma estratégia agressiva, um player adota metas ousadas de market share buscando lançar-se à frente de seus concorrentes para obter escala de produção suficiente para reduzir seus custos e conquistar retornos crescentes. Em uma estratégia conservadora, o player define um market share alvo modesto, devido à incerteza relacionada ao mercado e aceita dividir uma parcela maior de seu share com seus concorrentes, correndo menos risco de possuir capacidade excedente.

No modelo atual, a demanda global pelo produto é determinada em função do preço, e de parâmetros que estimam o tamanho do mercado potencial, e sua reação à acréscimos ou decréscimos no preço por meio de uma curva de preço versus demanda. A demanda global calculada obtida em equilíbrio com o preço é sujeita à um processo de difusão do produto. Considerar o processo de difusão de um novo produto é uma prática presente em diversos modelos similadres (Ex: Bass (XX), citar outros), visto que a difusão de um novo produto não é instantânea. A difusão do produto é dada a partir da demanda global determinada pelo preço, e parâmetros que medem a velocidade de difusão do produto no mercado alvo.

O próximo conjunto de módulos do modelo é vetorizado por produtores de impressora 3D (a partir deste

momento denominados como players). Esta característica torna o modelo útil para a avaliação da decisão estratégica de e um player específico, e permite a consideração de decisões estratégicas de outros players sobre o resultado da estratégia de um player em questão.Este aspecto será essencial para simular situações onde players existentes no mercado possuem estratégias de crescimento agressivas ou conservadoras, e o como estas decisões impactam o resultado da estratégia de um dado player. De modo similar, esta característica permite simular o impacto positivo que a expansão de outros players pode ter, expandindo o mercado de tal modo que haja mais demanda global para os demais players.

Este aspecto é relevante para a representação da indústria da manufatura aditiva, visto que a adição de capacidade por outros players, e decisões relacionadas à sua precificação tendem à influenciar a decisão da empresa.

Em seguida, a produção de cada um dos players simulados no moeolo é estimada, utilizando as informações de demanda, capacidade dos players e market share estimado. A produção, de modo imediato, gera caixa para os players, atualizando seu valor presente líquido em caixa.

Três macro-enlaces de feedback podem ser visualizados nesta estrutura. O primeiro enlace, R1, tende à estimular o crescimento da demanda por meio da expansão do mercado. Uma vez que parcelas cada vez maiores da

No modelo proposto por Sterman (XX) dois players, inicialmente com a mesma capacidade produtiva, iniciam vendendo produtos a um mercado em expansão.

8.2 Demanda Global

A demanda Total da indústria anual D^T é formada pela soma de dois tipos de demanda. A demanda inicial D^I dos produtos (ou seja, à primeira compra realizada por um usuário da impressora 3D), e à demanda oriúnda de recompras D^R, realizadas em função do fim da vida útil do equipamento.

$$D^T = D^I + D^R$$

A demanda inicial é calculada D^I em função do número médio de unidades vendidas por clientes μ e do número de clientes dA que adotou o produto em um intervalo de tempo dt:

$$D^I = \mu \frac{dA}{dt}$$

8.3 Difusão do Produto

O crescimento do número de clientes A que aderiram às impressoras 3D em um dado instante de tempo t é um estoque modelado por meio do modelo padrão de difusão de Bass (XXXX). Neste modelo o crescimento da população de clientes que aderem à uma ideia é dependente do tamanho total da população POP, do número de clientes que não adotaram N, da fração de inovadores que adotam ao produto ano a ano independentemente de outros usuários α e do parâmetro β que mede a força da difusão do produto por boca-a-boca. A não-negatividade da equação é garantida obtendo-se o máximo entre a equação e zero. Além disto, o valor inicial do número de clientes A_{t_0} é calibrado a partir do númer.....

$$A_t = A_{t_0} + \int_{t_0}^{t} MAX\left(0, N\left(\alpha + \beta \frac{A}{POP}\right)\right); A_{t_0} = \theta A^*$$

O número de consumidores potenciais N é modelado como o máximo entre zero e a diferença entre o número de clientes que irá adotar o produto em algum momento A^* e o número de clientes que adotou o produto A.

$$N = MAX(0, A^* - A)$$

O número de clientes que irá adotar o produto A^* é calculado segundo uma curva de demanda linear, variando em função do menor preço encontrado no mercado P^{min}, e da inclinação da curva de demanda σ, que corresponde à $(A^* - POP^r)/(P^{min} - P^r)$. Para a calibração da curva de preço e demanda, um preço de referência P^r e uma demanda de referência POP^r. Além disto, a demanda nunca será maior do que a população total POP, nem menor do que 0.

$$A^* = MIN\left(POP, POP^r * MAX\left(0, 1 + \frac{\sigma(P^{min} - P^r)}{POP^r}\right)\right)$$

A inclinação da curva de demanda σ, por sua vez, é calculada em função da população de referência POP^r, do preço de referência P^r e da elasticidade da curva de demanda ε_d.

$$\sigma = -\varepsilon_d\left(\frac{POP^r}{p^r}\right)$$

A demanda oriúnda da necessidade de substituição dos produtos depende do número de impressoras 3D já vendidos pela empresa I_i, e de uma taxa percentual de descarte de impressoras δ. Esta taxa percentual de descarte de impressoras corresponde ao inverso da vida útil média das impressoras vendidas. O modelo pressupõe que o número de impressoras descartadas pelo fim da sua vida útil corresponde ao número de impressoras a serem compradas.

$$D^r = \sum_i D_i \ ; \ D_i = \delta * I_i$$

Modificação: Este pressuposto atua como um pressuposto "otimista" para os produtores de impressoras 3D, implicando que, no longo prazo o mercado alcançado pelas impressoras 3D nunca retornarão à outras tecnologias. Podemos modificar esta equação incluindo uma taxa α de impressoras que são descartadas, porém nunca substituídas. Pensar numa forma de modelar esta taxa.

$$D^r = \sum_i D_i \ ; \ D_i = \delta * I_i * \alpha$$

O número de impressoras 3D atualmente instaladas em consumidores $I_{i,t}$ de cada player corresponde à acumulação de entregas de impressoras $I_{i,t}$ e é reduzida pelo número de produtos descartados $D_{i,t}$, considerando uma quantidade inicial I_{i,t_0} de impressoras instaladas no período inicial de simulação.

$$I_{i,t} = I_{i,t_0} + \int_{t_0}^{t} S_{i,t} - D_{i,t}$$

8.4 Market Share

A atratividade de cada player é calculada com base em um modelo logit de decisão (citar). Neste modelo, a atratividade de cada um dos players é calculada de acordo com um conjunto de critérios competitivos. No modelo de Sterman (XX), são utilizados como critérios o preço do produto e o tempo de entrega.

Modificação: Criar um módulo para estimar um índice de performance das impressoras 3D influenciados por investimentos em Pesquisa e Desenvolvimento. Desta maneira, o Share de cada produto player pode ser dividido de acordo com a performance dos diferentes players. Este módulo de performance do produto também pode ser influenciado pela curva de aprendizagem dos players.

$$A_i = exp\left(\varepsilon_p \frac{P_i}{P^r}\right) * exp\left(\varepsilon_a \left(\frac{B_i}{S_i}\right)/\tau^r\right)$$

Com base na atratividade de cada player, o market share é definido normalizando-se a atratividade dos players em conjunto. Esta formulação garante que a soma do market share de cada um dos players será igual a 1.

$$S_i = A_i / \sum_i A_i$$

Finalmente, os pedidos ganhos por cada empresa O_i são calculados de acordo com a Demanda Total da Indústria e de acordo com o seu share calculado.

$$O_i = S_i * D^T$$

8.5 A Firma

O lucro líquido a valor presente π_t da firma i é definido como um estoque calculado em função das receitas R_i e custos fixos C_i^f e variáveis C_i^v da empresa, trazidos a valor presente por um fator ρ. Desta maneira, o lucro líquido da empresa no tempo t será dado conforme esta equação:

$$\pi_t = \int_{t_0}^{t} [R_i - (C_i^f + C_i^v)] * e^{-\rho * t}$$

As receita bruta da empresa é calculada a partir do número de produtos entregues S_i pela empresa i e do preço médio de seus produtos vendidos $\bar{P}_i$, que é obtido pela divisão do valor da carteira de vendas V_i e de seu backlog B_i.

$$R_i = S_i * \bar{P}_i \; ; \bar{P}_i = \frac{V_i}{B_i}$$

O valor da carteira de vendas V_i aumenta conforme a quantidade de pedidos faturados $O_{i,t}$ e seu preço $P_{i,t}*$, e decresce à medida que produtos são entregues aos seus clientes gerando receita $R_{i,t}$.

$$V_{i,t} = V_{i,t_0} + \int_{t_0}^{t} P_{i,t} * O_{i,t} - R_{i,t}$$

Os custos fixos da empresa variam de modo proporcional à sua capacidade produtiva K_i, segundo um custo fixo unitário u_i^f. Os custos variáveis, por sua vez, são proporcionais ao número de produtos entregues pela empresa S_i, e um custo variável unitário u_i^v.

$$C_i^f = u_i^f * K_i \; ; \; C_i^v = u_i^v * S_i$$

Com o objetivo de demonstrar um mecanismo de retornos crescentes, Sterman (XX) insere em seu modelo um mecanismo de redução de custos oriúndo da curva de experiência. Esta formulação pressupõe que os players são capazes de reduzir seus custos à medida que produzem uma quantidade maior de produtos, obtendo experiência em produção E, equivalente dimensionalmente ao número de impressoras 3D produzidas. Os custos fixos u_i^f e variáveis u_i^v unitários caem à medida que a experiência E aumenta em relação à experiência inicial E_0.

$$u_i^f = u_0^f (E/E_0)^\gamma; \; u_i^v = u_0^v (E/E_0)^\gamma; \gamma = log(\Gamma)/log(2)$$

A amplitude desta redução é calibrada apartir de custos fixos e variáveis iniciais u_0^f e u_0^v, e de um parâmetro Γ que representa a força da curva de experiência. Esta formulação permite que os players em um primeiro

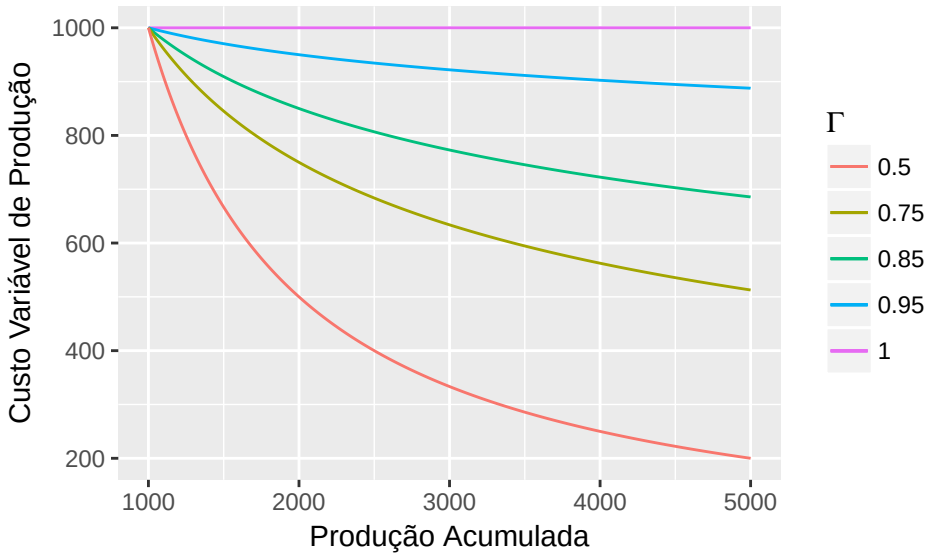


Figure 7: Relação entre Produção Acumulada e Custos

momento ampliem suas margens, e também permite que os mesmos reduzam seus preços com o objetivo de alcançar uma fatia maior de mercado. A Figura (XX) demonstra sensibilidade da relação não linear entre produção acumulada E e custos u_i^f e u_i^v, conforme varia a força da curva de experiência Γ.

Tal comportamento está em consonância com os dados observados na indústria da manufatura aditiva. Os preços das impressoras 3D tem caído expressivamente. Esta formulação, portanto, constitui-se como uma explicação estrutural para a queda dos preços nesta indústria. Esta formlua pressupõe que não há troca de experiência entre os players, e que não há "perda de experiência" de um determinado player.

A experiência $E_{i,t}$, por sua vez, é obtida a partir da acumulação da produção de cada player $E_{i,t}$.

Ponto para possível modificação. É possível imaginar um cenário onde, a longo prazo, a experiência obtida por um player é difundida para os demais players por meio de "cópias" e engenharia reversa. Seria possível imaginar uma forma de imaginar a uniformização do conhecimento, levando vantagens de curto prazo tenderem a se normalizar no longo prazo. Talvez seja mais inteligente levar essa ideia direto para o possível módulo de P & D.

Esta curva, sozinha, pode não explicar porque alguns players com menor volume de produção conseguem obter custos competitivos no mercado. Uma empresa talvez tenha apenas uma vantagem temporária em relação aos demais players. Este é um ponto a pensar, pois esta curva de experiência influencia os preços, que influenciam todos os demais comportamentos do modelo.

Este fator não "desmerece" o artigo do Sterman, visto que este pressuposto atua contra a hipótese dele, e o argumento que ele quiz usar não foi esse.

$$E_{i,t} = E_{i,t0} + \int_{t_0}^{t} S_i$$

8.5.1 Produção

O presente modelo diferencia a produção real da empresa, a produção desejada, e sua capacidade. Pressupõe-se que a empresa busca maximizar sua produção, logo sua produção corresponderá ao mínimo entre a sua capacidade produtiva e sua produção desejada. O modelo proposto por Sterman (XX) foi idealizado para representar decisões estratégicas de longo prazo, e não se dedicou a detalhar mecanismos de uma cadeia

de suprimentos à jusante ou à montante de cada um dos players. Por este motivo, a produção realizada pela empresa corresponde às entregas, desprezando a representação de estoques na cadeia de suprimentos da empresa. Considera-se a manutenção deste pressuposto adequada para os objetivos deste trabalho.

$$Q_i = MIN(Q_i^*, K_i); S_i = Q_i$$

Seguindo-se a lei de Little, o tempo médio de entrega corresponde à razão entre o backlog à taxa de entrega.

$$\tau_i = B_i/Q_i$$

O modelo pressupõe que cada uma das empresas possui um tempo de entrega alvo, ajustando sua taxa de produção à este tempo de entrega e ao backlog formado. Sendo assim, a taxa de produção alvo depende do backlog formado e desta taxa de entrega alvo.

$$Q_i^* = B_i/\tau_i^*$$

Por fim, o backlog de produção da empresa cresce com a chegada de pedidos e diminui com o envio de de produtos.

$$B_{i,t} = B_{i,t0} + \int_{t_0}^{t} O_i - Q_i$$

8.5.2 Capacidade

Neste modelo, a capacidade da empresa não pode se ajustar imediatamente à demanda. Sterman (XX) propõe a utilização do operador φ Erlang Lag, utilizado frequentemente para representar o delay embutido em processos de ajuste de capacidade (Sterman XX):

$$K_i = \varphi(K^*, \lambda)$$

A capacidade Alvo da Empresa K^*, por sua vez, é obtida a partir do market share alvo da empresa S^*, da demanda prevista para a indústria D^e e da taxa de utilização de capacidade u^*. A capacidade ainda é restrita a uma mínima escala de produção eficiente K^{min}.

$$K^* = MAX(K^{min}, S^* * D^e/u^*)$$

O modelo pressupõe que os players do mercado realizam estimativas de previsão de demanda λ anos à frente da demanda prevista com o objetivo de ajustar sua capacidade produtiva à demanda. Desta maneira, a demanda prevista D^e é estimada a partir da demanda reportada na indústria D^r e da taxa esperada de crescimento da demanda g^e. O modelo adota como pressuposto que as empresas extrapolam a demanda passada da indústria para prever a sua demanda futura.

$$D^e = D^r * exp(\lambda * g^e)$$

A taxa de crescimento da demanda, por sua vez, é estimada a partir de um horizonte histórico usado para a previsão h, comparando a reportada no período atual $D_t^r/$ e a demanda reportada no período $t-h$, D_{t-h}^r.

$$g^e = ln(D_t^r/D_{t-h}^r)/h$$

O modelo também admite que a empresa não possui a informação da demanda instantânea D^T. Desta maneira, a demanda reportada D^r não corresponde à demanda corrente, visto que há delays no processo de

comunicação do volume de vendas, mas sim ajusta-se à esta variável por meio de uma suavização exponencial de primeira ordem, conforme o parâmetro τ^r de suavização.

$$dD^r/dt = (D^T - D^r)/\tau^r$$

8.5.3 Estratégia de Capacidade da Firma

A variável de decisão criada no modelo de Sterman refere-se à estratégia de capacidade da firma. Sterman (XX) utiliza duas estratégias de capacidade distintas. Se a firma busca uma estratégia agressiva, a mesma busca um share dominante do mercado. Desta maneira a empresa define como o seu market-share alvo o máximo entre seu share mínimo desejado S_i^{min}, e o share que a empresa visualiza que outros players não atenderão S_i^u. Uma estratégia conservadora, por outro lado, define um market share máximo S_i^{max} que está disposta a ocupar no mercado. Caso a empresa observe que não haverá demanda suficiente para este market share em função de seus outros concorrentes, a empresa aceita como meta apenas o market share que outros players não atenderão S_i^u.

$$S^* = \begin{cases} MAX(S_i^{min}, S_i^u), \ if \ Str_i = Agress. \\ \\ MIN(S_i^{max}, S_i^u), \ if \ Str_i = Conserv. \end{cases}$$

O market share não disputado S_i^u é calculado em função da demanda não disputada D_i^u e da demanda prevista D^e.

$$S_i^u = MAX(0, D_i^u/D^e)$$

A demanda não contestada é obtida a partir da soma das capacidades de outros players esperada, da taxa de utilização da indústria e da demanda prevista.

$$D_i^u = D^e - u^* \sum_{j \neq i} K_j^e$$

A capacidade dos competidores esperada é obtida considerando que os players não possuem acesso à informação perfeita sobre o planejamento da capacidade dos outros players. Em um extermo, os demais players não tem nenhuma informação sobre a capacidade em construção dos outros players, e em outro extremo, os mesmos possuem informação perfeita sobre a capacidade em construção. O modelo utiliza um fator para expressar a parcela da capacidade em construção conhecida pelos demais players, permitindo que seja simulado o impacto desta variável sobre os resultados do modelo.

$$K_j^e = wK_j^{e^*} + (1-w)K_j$$

A capacidade alvo dos demais competidores é calculada considerando um delay de tempo, pressupondo que a empresa leva tempo para estimar e realizar os processos necessários para estimar a capacidade dos demais players.

$$dK_j^{e^*}/dt = (K_j^* - K_j^{e^*})/\tau^c$$

8.5.4 Preços

O modelo pressupõe que as empresas ajustam seus preços considerando seus custos unitários, a relação entre oferta e demanda e o seu market share atual e o market-share desejado. Na primeira parcela da equação, um preço base é calculado de acordo com os custos fixos e variáveis unitários, e de acordo com um markup desejado.

$$P^C = (1 + m^*)(u_i^f + u_i^f)$$

A partir deste preço base, a primeira parcelado preço alvo é calculada considerando a razão entre o preço base e o preço atual. Deste modo, se o preço base for maior do que o preço atual, a empresa tende a aumentar seus preços no futuro. A segunda parcela da equação relaciona a produção desejada da empresa com a sua capacidade efetiva, calculada a partir da sua taxa de utilização e sua capacidade. Novamente, se a produção desejada pela empresa é maior do que a sua capacidade, a empresa tende a aumentar seus preços, buscando otimizar a utilização de sua capacidade. Finalmente, a terceira parcela da equação utiliza a diferença entre o market share desejado pela empresa e seu market share atual. Deste modo, se o market share da empresa for menor do que o market share desejado, a empresa tende a reduzir seu preço, para alcançar o market share desejado.

$$P_i^* = MAX\left[u_i^v, P_i\left(1 + \alpha^c\left(\frac{P_i^c}{P_i} - 1\right)\right)\left(1 + \alpha^d\left(\frac{Q_i^*}{u_i^* K_i} - 1\right)\right)\left(1 + \alpha^s\left(S_i^* - S_i\right)\right)\right]$$

Em uma situação onde o preço atual é igual ao preço base, a produção desejada é igual à capacidade efetiva, e o market share atual é igual ao market share desejado, não realizará mudanças em seu preço. Caso qualquer uma destas igualdades não seja satisfeita, a empresa mudará seu preço alvo para um novo valor. Além disto, o modelo pressupõe que as empresas do modelo não precificarão seus produtos abaixo do custo variável.

A partir do preço alvo calculado, pressupõe-se que processos burocráticos não permitem que as empresas ajustem seu preço instantâneamente. Desta maneira, obtém-se o preço praticado pelos players por meio de uma suavização exponencial de primeira ordem, considerando um tempo de ajuste.

$$dP_i/dt = (P_i^* - P_i)/\tau^p$$

8.5.5 Pesquisa e Desenvolvimento e Patentes

O investimento em Pesquisa e Desenvolvimento é representado por uma decisão da empresa relacionada à fração de sua receita a ser investida. Este investimento é acumulado em um estoque, representando o volume de investimentos que ainda não gerou patentes.

$$D_i/dt = (Receita * Orc) - D_i/TempoMédio$$

8.6 Síntese das Modificações Realizadas

Table 4: Modificações Realizadas em Relação ao Modelo Original

Módulo	**Necessidade de Modificação**	**Modificação Realizada**
Market Share	Market Share é apenas dividido por preço e delay na entrega, enquanto a performance do produto não parece ser considerada.	Criar setor de investimento em P&D influenciando a performance do produto juntamente com a experiência de produção.

Módulo	Necessidade de Modificação	Modificação Realizada
Capacidade	Estratégia de crescimento é "Conservadora" ou "Agressiva", e não possui opção adaptativa.	Avaliar primeiro as duas estratégias na primeira rodada do modelo e em seguida adicionar uma estratégia adaptativa (provavelmente a agressiva no início e conservadora no final).
Parâmetros	Parâmetros possuem valores iniciais não aderentes à indústria da manufatura aditiva.	Modificar parâmetros e calibrar modelo para a manufatura aditiva.
Número de Players	Modelo original considera apenas dois players	Modificar para 10 players (considerar os 9 maiores players e agregar os demais em um player "outros".)

8.7 Implementação do Modelo Computacional

O modelo matemático descrito na seção anterior foi implementado no software R. O código fonte implementado no software R está disponível no Apêndice (XX). Adicionalmente, o modelo foi implementado no software Ithink 10.0.3, com o propósito de verificar a consistência dos resultados obtidos no software R. Considerando que a integração numérica invariavelmente traz erros ao processo do calculo, (Sterman XXX). O modelo foi implementado segundo as diretrizes constantes em Dungan (XXXX), e utilizou a biblioteca deSolve (procurar e Citar XXXX) para a resolução das equações diferenciais.

Os resultados deste trabalho podem ser observados no link bit.ly/reproddissertpnl.

8.7.1 Testes Estruturais / Testes de Valores Extremos

8.8 Calibração do Modelo e Comparação com Dados Históricos

A modelagem exploratória, per si, abandona a premissa de que modelos de simulação computacional apenas serão úteis se validados experimentalmente (Bankes XX). Ainda assim, os modelos podem ser verificados visando avaliar sua consistência interna, bem como os seus resultados podem ser comparados com dados históricos para observar a capacidade do modelo em explicar o comportamento passado. (Sterman XXXX) Considerando estas premissas, esta seção apresenta os testes e calibrações realizadas no modelo.

O objetivo deste procedimento de calibração foi ajustar os dados de entrada do modelo à ordem de grandeza dos números observados em empresas da indústria da manufatura aditiva. Neste sentido, um conjunto de dados financeiros foi coletado com o propósito de comparar tais dados aos resultados da simulação.

Considerando a característica estratégica da maior parte dos dados relacionados ao modelo, foi necessário recorrer à bases de informação públicamente disponíveis. Sendo assim, foi realizada uma pesquisa junto à websites que possuíram dados financeiros relacionados à industria.

8.9 Tabela de Parâmetros Calibrados

Table 5: Parâmetros Incertos e Unidades de Medidas

Parâmetro	Significado	Unidade	Nome Interno
μ	Número de Unidades Vendidas por Consumidor	Produto / Consumidores	aUnitsPerHousehold
ρ	Taxa de Disconto	adimensional	aDiscountRate
ρ	Taxa de Disconto	adimensional	aDiscountRate

Table 6: Parâmetros e Valores de Referência

Parâmetro	S CB	S Min	S Max	CB	Min	Max
μ	1	1	1	1	0,5	3

Calibração da Demanda Global:

Variáveis utilizadas na calibração e erros:

Tabela de Resíduos da Calibração:

Parâmetros calibrados com este procedimento:

8.10 Análise RDM

8.10.1 XLRM

8.10.2 Geração de Casos

8.10.2.1 Geração inicial de Casos Plausíveis

Estas são as opções Iniciais utilizadas para a geração dos casos plausíveis.

```
## $VarResposta
## [1] "sNPVProfit1"
##
## $VarCenarios
## [1] "Scenario"
##
## $VarEstrategias
## [1] "Lever"
##
## $N
## [1] 100
##
## $VarTempo
## [1] "time"
##
## $VarCriterio
## [1] "RegretPercPercentil75"
##
## $SentidoCriterio
## [1] "min"
##
## $Paralelo
## [1] TRUE
##
## $ModoParalelo
## [1] "FORK"
##
## $SimularApenasCasoBase
## [1] TRUE
##
## $FullFactorialDesign
```

```
## [1] TRUE
```

- Parâmetros dos Casos Plausíveis (primeira rodada).

Variavel	NomeAmigavel
aUnitsPerHousehold	Unidades por Consumidor
aDiscountRate	Taxa de Desconto
aNormalDeliveryDelay	Tempo de Entrega Padrão
aSwitchForCapacity	Configuração: Capacidade influencia a produção ou não.
aFractionalDiscardRate	Percentual de Produtos Descartados
aInitialDiffusionFraction	Fração Inicial de Difusão dos Produtos
aReferencePrice	Preço de Referência em Equilíbrio com Demanda de Referência
aReferenceIndustryDemandElasticity	Elasticidade da Demanda de referência
aReferencePopulation	Mercado Consumidor de Referência
aInnovatorAdoptionFraction	Fração de Consumidores Inovadores
aWOMStrength	Força da Difusão do Produto “Boca a Boca”
aPopulation	Número Total de Consumidores no modelo
aSwitchForShipmentsInForecast	Configuração: Entregas Influencia a Produção.
aVolumeReportingDelay	Tempo de delay de report da demanda
aForecastHorizon	Horizonte de Previsão
aCapacityAcquisitionDelay	Delay no tempo de aquisição de capacidade
aTimeForHistoricalVolume	Tempo de coleta de dados históricos
aReferenceDeliveryDelay	Tempo padrão de entrega
aSensOfAttractToAvailability	Sensibilidade da atratividade ao tempo de entrega
aSensOfAttractToPrice	Sensibilidade da atratividade ao preço
aLCStrength	Força da Curva de Aprendizagem
aInitialProductionExperience	Experiência de Produção Inicial
aRatioOfFixedToVarCost	Razão dos Custos Fixos em relação aos custos variáveis
aNormalProfitMargin	Margem de Lucro padrão
aNormalCapacityUtilization	Utilização da Capacidade padrão
aMinimumEfficientScale	Escala mínima de eficiência
aWeightOnSupplyLine	?
aTimeToPerceiveCompTargetCapacity	Tempo de reconhecimento da capacidade alvo de outros players.
aPriceAdjustmentTime	Tempo de delay do ajuste de preços.
aSensOfPriceToCosts	Sensibilidade do Preço aos Custos
aSensOfPriceToDSBalance	Sensibilidade do preço à Oferta e Demanda
aSensOfPriceToShare	Sensibilidade do Preço ao Market-Share
aSwitchForPerfectCapacity	Configuração: Capacidade Perfeita
aPeDLigado	Configuração: PeD Ligado
aTempoMedioRealizacaoPeD	Tempo Médio para um investimento em PeD gerar uma patente.
aCustoMedioPatente	Custo médio de obtenção de uma patente.
aTempoMedioAvaliacao	Tempo Médio para a rejeição ou concessão de uma patente.
aTaxaRejeicao	Percentual de patentes solicitadas que são rejeitadas.
aTempoVencimentoPatentes	Tempo de Expiração de uma patente.
aTempodeInutilizacaoPatente	Tempo de Inutilização (após a expiração de uma patente).
aPerfSlope	Melhoria em performance por patente que a empresa tem acesso.
aPerfMin	Índice de Performance Mínimo
aPerfMax	Índice de Performance Máximo
aSensOfAttractToPerformance	NA
aReferencePerformance	NA
aInitialInvestimentoNaoRealizadoPeD	NA
aInitialPatentesRequisitadas	NA
aInitialPatentesEmpresa	NA
aInitialsPatentesEmDominioPublicoUteis	NA

Variavel	NomeAmigavel
aInitialsInvestimentoPeDDepreciar	NA
aInitialReorderShare	NA
aTotalInitialInstalledBase	NA
aInitialIndustryShipments	NA
aInitialSharePlayers1	NA
aInitialSharePlayers2	NA
aInitialSharePlayers3	NA
aInitialSharePlayers4	NA
aPatentShare1	Share de Patentes
aPatentShare2	NA
aPatentShare3	NA
aPatentShare4	NA
aPercPeDAberto2	Percentual do Orçamento de PeD aplicado ao desenvolvimento de tecnologia ser
aPercPeDAberto3	Percentual do Orçamento de PeD aplicado ao desenvolvimento de tecnologia ser
aPercPeDAberto4	Percentual do Orçamento de PeD aplicado ao desenvolvimento de tecnologia ser
aOrcamentoPeD2	Percentual de Orçamento Direcionado a PeD
aOrcamentoPeD3	Percentual de Orçamento Direcionado a PeD
aOrcamentoPeD4	Percentual de Orçamento Direcionado a PeD
aDesiredMarketShare2	Market Share Desejado* Na Estratégia Agressiva (talvez seja melhor separar)
aDesiredMarketShare3	Market Share Desejado* Na Estratégia Agressiva (talvez seja melhor separar)
aDesiredMarketShare4	Market Share Desejado* Na Estratégia Agressiva (talvez seja melhor separar)
aSwitchForCapacityStrategy2	Estratégia de Capacidade
aSwitchForCapacityStrategy3	Estratégia de Capacidade
aSwitchForCapacityStrategy4	Estratégia de Capacidade
aInitialPrice1	Preço de Referência Inicial
aInitialPrice2	Preço de Referência Inicial
aInitialPrice3	Preço de Referência Inicial
aInitialPrice4	Preço de Referência Inicial

- Ensemble Gerado:

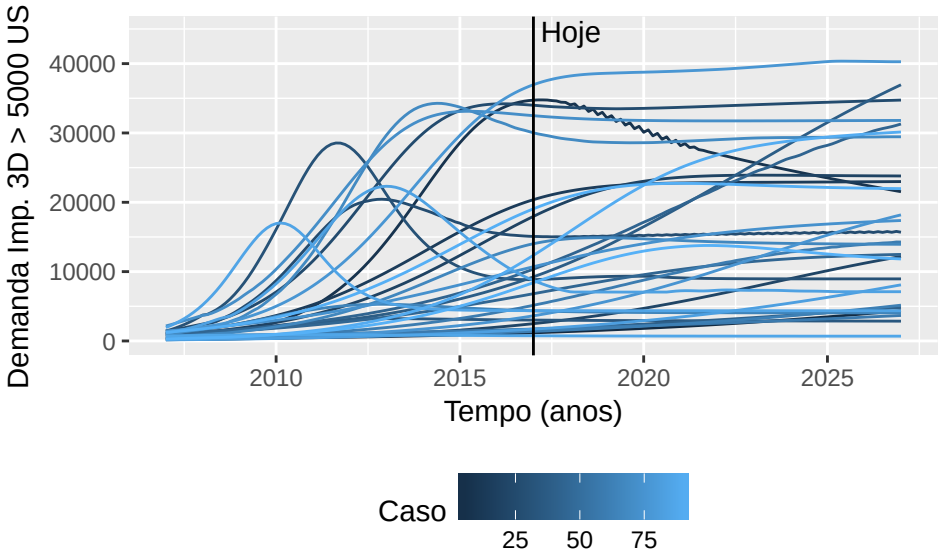
Scenario	aUnitsPerHousehold	aDiscountRate	aNormalDeliveryDelay	aSwitchForCapacity	aFractionalDiscardRate
1	1	0.04	0.25	1	0.3033792
2	1	0.04	0.25	1	0.3296901
3	1	0.04	0.25	1	0.3157938
4	1	0.04	0.25	1	0.3733956
5	1	0.04	0.25	1	0.5830289
6	1	0.04	0.25	1	0.2773220
7	1	0.04	0.25	1	0.1350045
8	1	0.04	0.25	1	0.4437441
9	1	0.04	0.25	1	0.1445513
10	1	0.04	0.25	1	0.1059189

- Dados Simulados:

time sNP	VProfit1 sNP	VProfit2 sNP	VProfit3 sNP	VProfit4 sVa	lueOfBacklog1 sVa	lueOfBacklog2 sVa	lueO
2007.000	0.0	0.0	0.00	0.0	3510039	3510039.0	1.17
2007.125	388275.7	384691.2	129266.04	383960.9	3705015	3705014.8	5.85
2007.250	853926.6	846577.2	20125.67	845080.0	3902182	3718677.8	2.92
2007.375	1397192.3	1315292.3	-206750.23	1406241.6	4115206	3665477.0	1.46
2007.500	2023163.6	1839367.2	-472249.25	2036925.2	4312534	3669331.0	7.34

time sNP	VProfit1 sNP	VProfit2 sNP	VProfit3 sNP	VProfit4 sVa	lueOfBacklog1 sVa	lueOfBacklog2 sVa	lueO
2007.625	2691287.2	2492315.3	-733366.39	2677679.8	4486282	3724944.1	3.71
2007.750	3357049.9	3140570.8	-971227.81	3288903.4	4927768	3380195.2	1.91
2007.875	4104418.5	3669438.0	-1180747.08	3975868.0	5451828	2817699.6	1.03
2008.000	4953421.3	4096490.2	-1363653.26	4776330.8	5864818	2341569.5	6.20
2008.125	5856073.4	4442345.8	-1524355.86	5644207.9	6208128	1967588.2	4.24
2008.250	6785682.7	4724992.3	-1667760.62	6543691.1	6507561	1664969.3	3.26
2008.375	7727234.7	4960036.4	-1798297.22	7448518.1	6769848	1423643.8	2.66
2008.500	8668845.2	5160800.6	-1919564.82	8341376.5	6998496	1243325.3	2.22
2008.625	9600298.7	5337908.5	-2034293.20	9214123.4	7191801	1118368.7	1.82
2008.750	10514549.8	5499160.4	-2144465.29	10065207.8	7360040	1036987.3	1.43
2008.875	11410723.0	5649878.0	-2251479.59	10897997.5	7517449	984706.6	1.03
2009.000	12292949.6	5793538.6	-2356298.07	11717718.7	7674003	951093.6	6.54
2009.125	13167512.2	5932430.0	-2459559.97	12529515.7	7834089	930180.9	3.55
2009.250	14040382.4	6068068.4	-2561662.80	13337886.8	7998946	917829.6	1.78
2009.375	14915976.6	6201445.0	-2662838.96	14146718.6	8168456	911063.6	8.93

- Curvas de Demanda Simuladas (Antes de Filtrar os Casos).



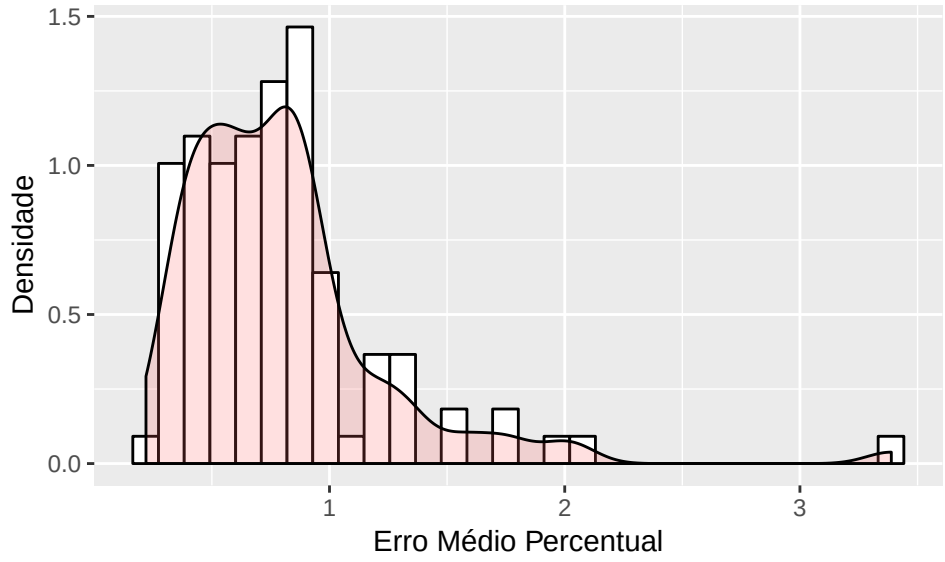
8.10.2.2 Comparação das Simulações com Série Histórica

Exemplo de 5 Casos Comparados com o Histórico:

	1	2	3	4	5
Scenario	1.000000e+00	2.000000e+00	3.000000e+00	4.000000e+00	5.000000e+00
SumOfSquareResiduals	4.211294e+08	4.753098e+08	6.036684e+09	1.106047e+08	2.406467e+08
MeanSquareError	3.828449e+07	4.320998e+07	5.487895e+08	1.005497e+07	2.187697e+07
MeanAbsoluteError	5.597142e+03	5.943433e+03	2.004411e+04	2.823234e+03	4.342785e+03
MeanAbsolutePercentError	9.030375e-01	9.576449e-01	3.388417e+00	4.574918e-01	7.255123e-01
UM_ThielBiasDiffMeans	8.182950e-01	8.175053e-01	7.199500e-01	7.927073e-01	8.620837e-01
US_ThielUnequalVariation	1.983037e-01	1.999746e-01	2.197468e-01	1.915277e-01	1.435518e-01
UC_ThielUnequalCovariation	1.571800e-03	7.695000e-04	8.830820e-02	3.649430e-02	8.156100e-03

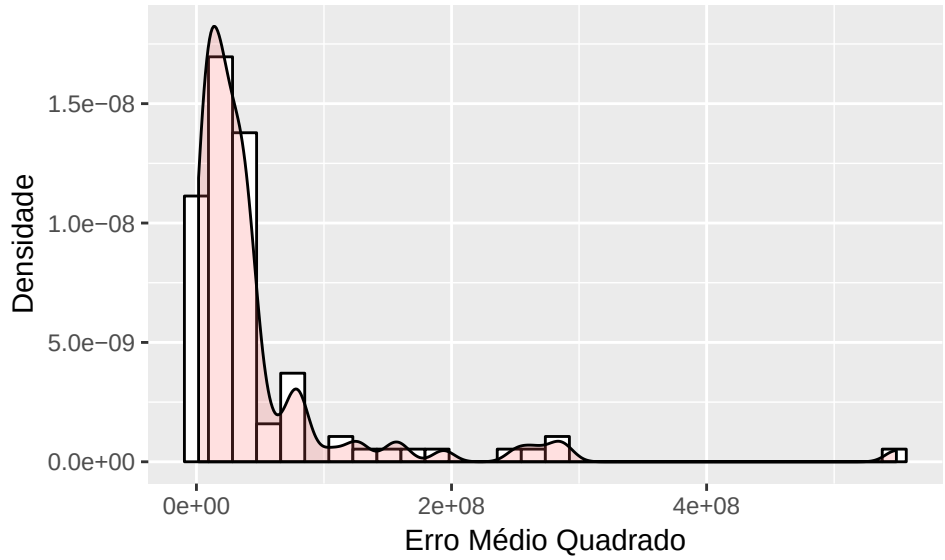
- Exibindo Erro dos Casos Simulados em Relação aos Dados históricos (erro médio percentual).

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



- Exibindo Erro dos Casos Simulados em Relação aos Dados históricos (erro médio quadrado).

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



- Identificando o Cenário Simulado com o Menor Erro Médio Quadrado (MSE):

```
## [1] 73
```

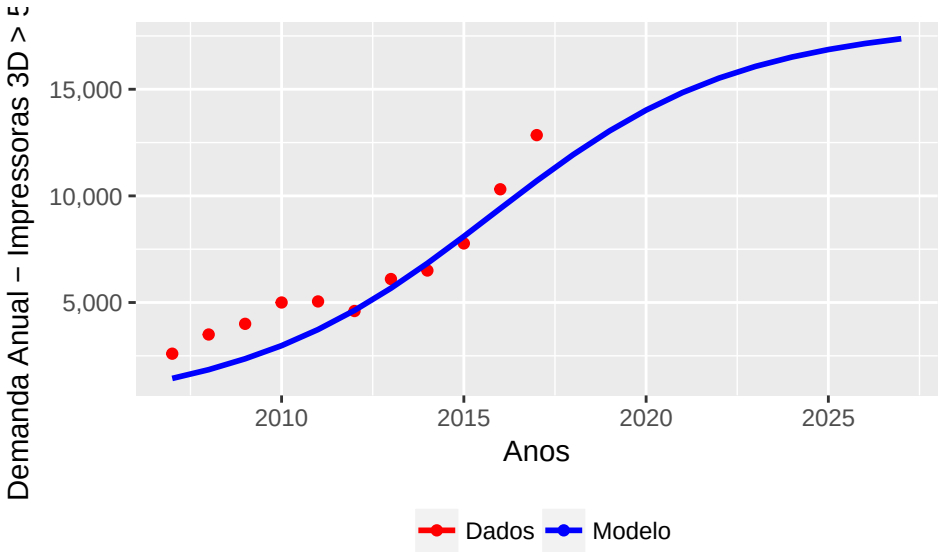
- Observando os Parâmetros deste Cenário:

	73
Scenario	7.300000e+01
aUnitsPerHousehold	1.000000e+00
aDiscountRate	4.000000e-02
aNormalDeliveryDelay	2.500000e-01
aSwitchForCapacity	1.000000e+00

	73
aFractionalDiscardRate	5.779561e-01
aInitialDiffusionFraction	5.000000e-02
aReferencePrice	2.000000e+05
aReferenceIndustryDemandElasticity	2.753285e-01
aReferencePopulation	3.242559e+04
aInnovatorAdoptionFraction	7.578000e-04
aWOMStrength	4.271600e-01
aPopulation	4.839113e+04
aSwitchForShipmentsInForecast	0.000000e+00
aVolumeReportingDelay	3.595798e-01
aForecastHorizon	1.000000e+00
aCapacityAcquisitionDelay	8.003785e-01
aTimeForHistoricalVolume	1.000000e+00
aReferenceDeliveryDelay	2.500000e-01
aSensOfAttractToAvailability	-2.222815e+00
aSensOfAttractToPrice	-8.992022e+00
aLCStrength	6.954182e-01
aInitialProductionExperience	1.000000e+05
aRatioOfFixedToVarCost	3.099045e+00
aNormalProfitMargin	2.344094e-01
aNormalCapacityUtilization	7.128226e-01
aMinimumEfficientScale	1.000000e+01
aWeightOnSupplyLine	1.000000e+00
aTimeToPerceiveCompTargetCapacity	2.500000e-01
aPriceAdjustmentTime	2.500000e-01
aSensOfPriceToCosts	1.361505e+00
aSensOfPriceToDSBalance	1.845738e-01
aSensOfPriceToShare	-1.018111e-01
aSwitchForPerfectCapacity	0.000000e+00
aPeDLigado	1.000000e+00
aTempoMedioRealizacaoPeD	3.038852e+00
aCustoMedioPatente	1.000000e+05
aTempoMedioAvaliacao	1.690234e+00
aTaxaRejeicao	3.123363e-01
aTempoVencimentoPatentes	1.800000e+01
aTempodeInutilizacaoPatente	1.117025e+01
aPerfSlope	4.037460e-02
aPerfMin	0.000000e+00
aPerfMax	1.000000e+01
aSensOfAttractToPerformance	-3.955759e+00
aReferencePerformance	1.000000e+01
aInitialInvestimentoNaoRealizadoPeD	2.000000e+03
aInitialPatentesRequisitadas	2.000000e+02
aInitialPatentesEmpresa	2.000000e+02
aInitialsPatentesEmDominioPublicoUteis	2.000000e+01
aInitialsInvestimentoPeDDepreciar	1.000000e+06
aInitialReorderShare	6.122118e-01
aTotalInitialInstalledBase	2.500000e+03
aInitialIndustryShipments	1.831000e+03
aInitialSharePlayers1	3.000000e-01
aInitialSharePlayers2	3.000000e-01
aInitialSharePlayers3	1.000000e-01

	73
aInitialSharePlayers4	3.000000e-01
aPatentShare1	3.000000e-01
aPatentShare2	3.000000e-01
aPatentShare3	1.000000e-01
aPatentShare4	3.000000e-01
aPercPeDAberto2	5.358329e-01
aPercPeDAberto3	6.488351e-01
aPercPeDAberto4	9.632652e-01
aOrcamentoPeD2	2.187800e-03
aOrcamentoPeD3	3.860900e-03
aOrcamentoPeD4	3.254200e-03
aDesiredMarketShare2	2.394548e-01
aDesiredMarketShare3	5.266740e-02
aDesiredMarketShare4	3.577180e-01
aSwitchForCapacityStrategy2	5.386650e-01
aSwitchForCapacityStrategy3	5.963210e-02
aSwitchForCapacityStrategy4	4.419245e-01
aInitialPrice1	2.000000e+05
aInitialPrice2	2.000000e+05
aInitialPrice3	2.000000e+05
aInitialPrice4	2.000000e+05
SumOfSquareResiduals	1.841151e+07
MeanSquareError	1.673773e+06
MeanAbsoluteError	1.087674e+03
MeanAbsolutePercentError	2.198857e-01
UM_ThielBiasDiffMeans	5.520366e-01
US_ThielUnequalVariation	5.619400e-03
UC_ThielUnequalCovariation	4.871404e-01

- Comparando Dados Simulados e Histórico Observado:



8.10.2.3 Definindo Casos a Simular

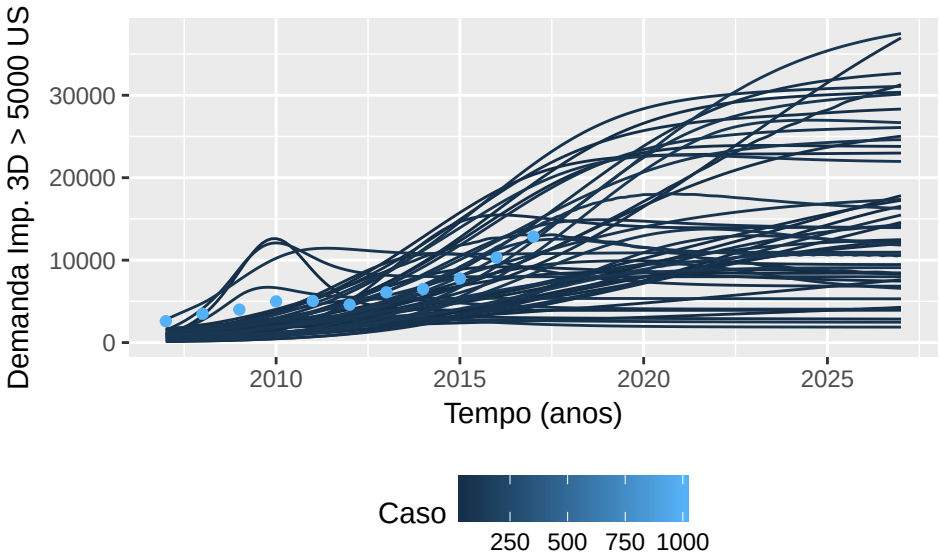
- Definindo Casos Considerados plausíveis

critério: A metade dos casos com menor Soma do Erro Médio Quadrado.

```
## PercentilCriterio
##               0.5
```

```
##       50%
## 274380905
```

- Definindo casos Considerados plausíveis (aqueles com erro quadrado médio menor do que o percentil). Exibindo casos:



- Exibindo propriedades do ensemble a simular:

```
##  num [1:50, 1:78] 4 5 7 9 11 14 17 19 20 22 ...
##  - attr(*, "dimnames")=List of 2
##   ..$ : NULL
##   ..$ : chr [1:78] "Scenario" "aUnitsPerHousehold" "aDiscountRate" "aNormalDeliveryDelay" ...
```

8.10.2.4 Simulando o Comportamento de Estratégias nos Casos

- Observando Combinações de Estratégias a Simular:

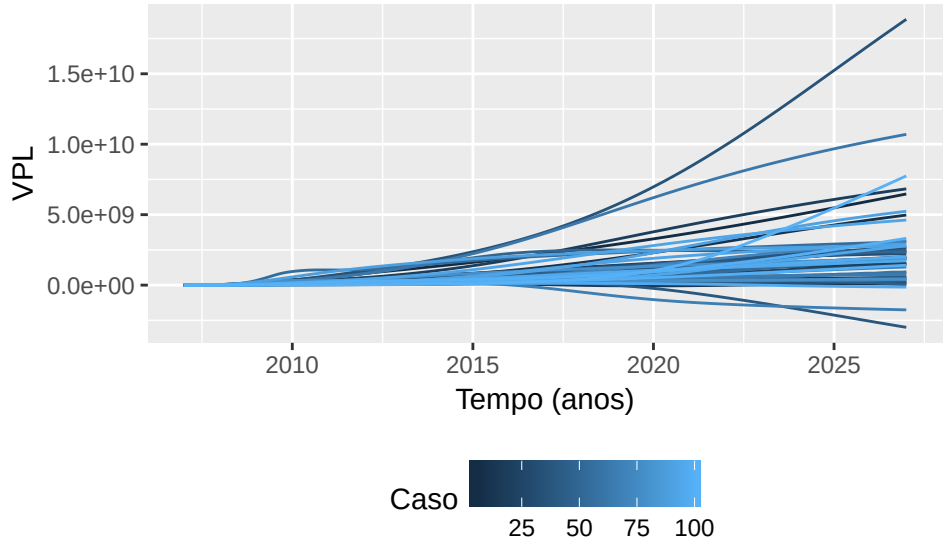
aSwitchForCapacityStrategy1	1.000	NA	NA
aPercPeDAberto1	0.000	0.5	1.0
aDesiredMarketShare1	0.300	0.5	0.7
aOrcamentoPeD1	0.001	0.1	0.3

- Projeto de Experimentos: Fatorial Completo. Gera-se o quadro de estratégias abaixo.

Lev	er Leve	rCode Cas	oBase aSw	itchForCapacityStrategy1 aPe	rcPeDAberto1 aDe	siredMarketShare1 aOr	c
1	1	1	1	1	0.0	0.3	0
4	2	2	0	1	0.5	0.3	0
7	3	3	0	1	1.0	0.3	0
10	4	4	0	1	0.0	0.5	0
13	5	5	0	1	0.5	0.5	0
16	6	6	0	1	1.0	0.5	0
19	7	7	0	1	0.0	0.7	0
22	8	8	0	1	0.5	0.7	0

Lev	er Leve	rCode Cas	oBase aSw	itchForCapacityStrategy1 aPe	rcPeDAberto1 aDe	siredMarketShare1 aOr	c
25	9	9	0	1	1.0	0.7	0
28	10	10	0	1	0.0	0.3	0
31	11	11	0	1	0.5	0.3	0
34	12	12	0	1	1.0	0.3	0
37	13	13	0	1	0.0	0.5	0
40	14	14	0	1	0.5	0.5	0
43	15	15	0	1	1.0	0.5	0
46	16	16	0	1	0.0	0.7	0
49	17	17	0	1	0.5	0.7	0
52	18	18	0	1	1.0	0.7	0
55	19	19	0	1	0.0	0.3	0
58	20	20	0	1	0.5	0.3	0
61	21	21	0	1	1.0	0.3	0
64	22	22	0	1	0.0	0.5	0
67	23	23	0	1	0.5	0.5	0
70	24	24	0	1	1.0	0.5	0
73	25	25	0	1	0.0	0.7	0
76	26	26	0	1	0.5	0.7	0
79	27	27	0	1	1.0	0.7	0

- Exibindo Resultados das Simulações



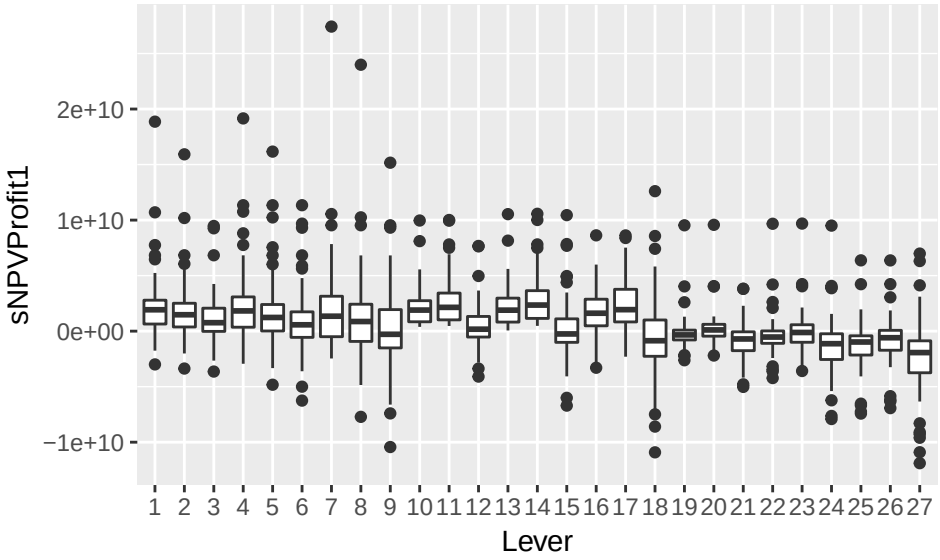
8.10.2.5 Comparação das Estratégias - Análise da Perda de Oportunidade

- Tabela de Análise da Perda de Oportunidade

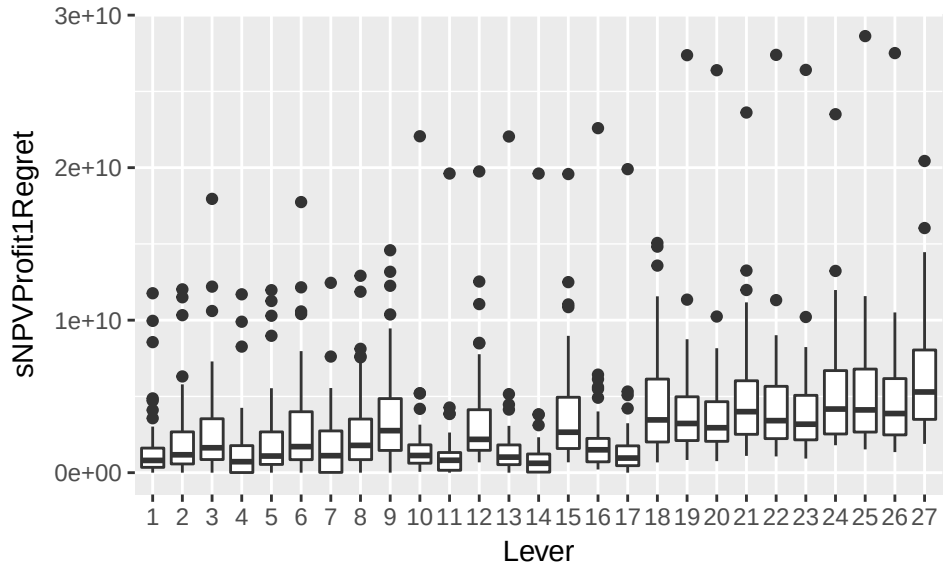
Lever	sNPVProfit1Medio	sNPVProfit1Desvio	sNPVProfit1Percentil25	sNPVProfit1Percentil75	sNPVProfit1Regre
1	2482777438	3308758110	640463141	2788458074	1655
2	1923680249	3066628076	376625660	2511955287	2214
3	1217560818	2477597452	-21712308	2068320333	2921
4	2587141442	3760815561	354690144	3084143689	1551
5	1856511518	3700204598	15436682	2399587855	2282
6	1005007005	3386441620	-558333875	1738780679	3133

Lever	sNPVProfit1Medio	sNPVProfit1Desvio	sNPVProfit1Percentil25	sNPVProfit1Percentil75	sNPVProfit1Regre
7	2257507777	4717370772	-500015550	3144555621	1881
8	1328982291	4690966450	-923971067	2425572823	2809
9	277995761	4324304769	-1510363986	1941212561	3860
10	2347061564	1988454759	843583815	2743279938	1791
11	2803388883	2349972469	1016987677	3428267479	1335
12	522855189	2236847265	-531484370	1323215043	3615
13	2403000463	2128691442	816492642	2972450549	1735
14	2945462442	2434998720	1147092708	3644759397	1193
15	240618371	3194903500	-1008220279	1107885064	3898
16	1854294884	2359457134	479472974	2876396367	2284
17	2488141686	2536956016	831129699	3767958601	1650
18	-592165884	4128868545	-2264634075	1006384681	4730
19	-85217031	1770671370	-791430621	94182163	4223
20	228142224	1786398306	-452441506	614928883	3910
21	-900975529	1803719977	-1756984457	-61092398	5039
22	-388726636	2039512349	-1096371961	20818962	4527
23	-18209072	2021531419	-1001501684	575227394	4156
24	-1279568503	2949142493	-2552281274	-238096800	5418
25	-1225428803	2421405918	-2151518396	-421253307	5364
26	-824584708	2342165345	-1719373164	75457966	4963
27	-2401621355	3762086111	-3748196552	-871872026	6540

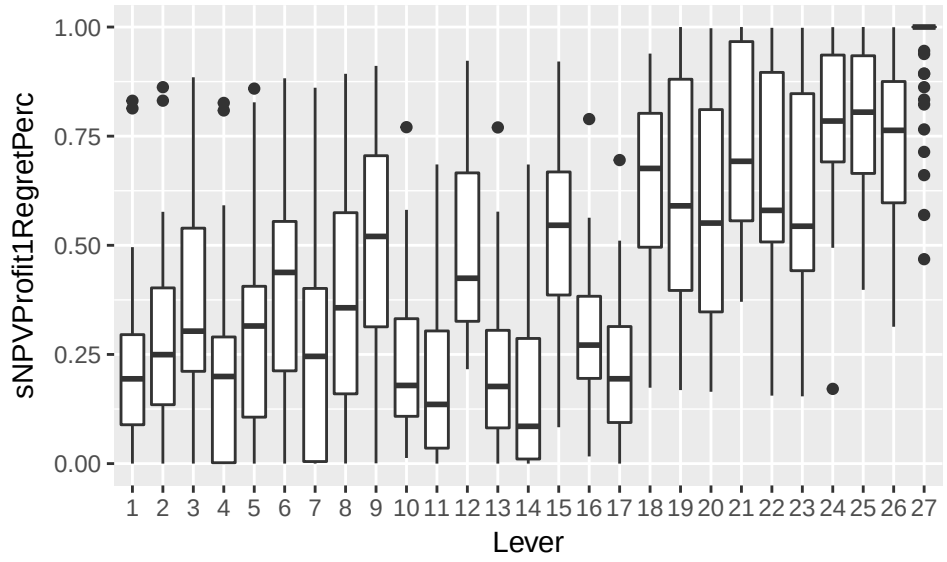
- Gráficos para Comparação das Estratégias - NPV Profit:



- Gráficos para Comparação das Estratégias - NPV Regret:



- Gráficos para Comparação das Estratégias - NPV Regret Percentual:



8.10.3 Análise de Vulnerabilidades

8.10.4 Modificações do Modelo para a segunda Rodada

8.10.5 Geração de Casos (Rodada 2)

8.10.6 Análise de Vulnerabilidades (Rodada 2)

8.10.7 Análise de Tradeoffs

8.11 Discussão dos Resultados

9 Conclusões

10 Apêndices

10.1 Código Fonte da Ferramenta Computacional

```
## expression(library(plotly), library(lhs), library(deSolve), library(dplyr),
##     library(ggplot2), library(GGally), library(viridis), library(season),
##     library(gridExtra), library(akima), library(parallel), library(FME),
##     library(readxl), library(gdata), library(scales), library(Quandl),
##     VAR_SCENARIO = "Scenario", VAR_LEVER = "Lever", SIM_TIME,
##     solve_modelo_dissertacao <- function(parametros, modelo, simtime){
##
##       # Número de Players no modelo
##       N_PLAYERS <<- 4
##
##       # All the stocks are initialised here...
##
##       n_tempo = length(simtime)
##
##       list.variaveis.globais <<- list(
##         sReportedIndustryVolume = matrix(NA, ncol = N_PLAYERS, nrow = n_tempo),
##         aExpectedIndustryDemand = matrix(NA, ncol = N_PLAYERS, nrow = n_tempo)
##       )
##
##       ordem_vetores_players = order(names(parametros[grep("aSwitchForCapacityStrategy", x = names(pa
##
##       ##### VARIÁVEIS DE ENTRADA - AUXILIARES #####
##       auxs    <- list(
##                       # Variáveis informadas de modo Independente por Player:
##                       # Estratégia de Capacidade:
##                       # A ordem dos números a seguir obedece a ordem na planilha e a ordem na geraçã
##                       aSwitchForCapacityStrategy = unname(round(parametros[grep("aSwitchForCapacityS
##                       ,aDesiredMarketShare = unname(parametros[grep("aDesiredMarketShare", x = names
##                       # Variáveis de Decisão - Existem para o player analisado e para os outros Play
##                       ,aOrcamentoPeD =  unname(parametros[grep("aOrcamentoPeD", x = names(parametros
##                       ,aPercPeDAberto =  unname(parametros[grep("aPercPeDAberto", x = names(parametr
##                       # A Initial Price
##                       ,aInitialPrice = unname(parametros[grep("aInitialPrice", x = names(parametros)
##                       ,aPatentShare = unname(parametros[grep("aPatentShare", x = names(parametros))]
```

```
##                         ,aInitialSharePlayers = unname(parametros[grep("aInitialSharePlayers", x = name
##
##
##                         # Outras Variáveis.
##                         ,aDiscountRate = unname(parametros["aDiscountRate"])
##                         ,aNormalDeliveryDelay = rep(unname(parametros["aNormalDeliveryDelay"]), times =
##                         ,aSwitchForCapacity = unname(parametros["aSwitchForCapacity"])
##                         # Vamos testar apenas um parâmetro por enquanto
##                         ,aFractionalDiscardRate = unname(parametros["aFractionalDiscardRate"]) # unname
##                         ,aInitialDiffusionFraction = unname(parametros["aInitialDiffusionFraction"])
##                         ,aReferencePrice = unname(parametros["aReferencePrice"])
##                         ,aReferenceIndustryDemandElasticity = unname(parametros["aReferenceIndustryDema
##                         ,aReferencePopulation = unname(parametros["aReferencePopulation"])
##                         ,aInnovatorAdoptionFraction = unname(parametros["aInnovatorAdoptionFraction"])
##                         ,aWOMStrength = unname(parametros["aWOMStrength"]) # unname(pars["aWOMStrength
##                         ,aPopulation = unname(parametros["aPopulation"]) #100000000 # Original Sterman
##                         ,aUnitsPerHousehold = unname(parametros["aUnitsPerHousehold"])
##                         ,aSwitchForShipmentsInForecast = unname(parametros["aSwitchForShipmentsInForeca
##                         ,aVolumeReportingDelay = rep(unname(parametros["aVolumeReportingDelay"]), times
##                         ,aForecastHorizon = rep(unname(parametros["aForecastHorizon"]), times = N_PLAY
##                         ,aCapacityAcquisitionDelay = unname(parametros["aCapacityAcquisitionDelay"])
##                         ,aTimeForHistoricalVolume = unname(parametros["aTimeForHistoricalVolume"])
##                         # Market Sector
##                         ,aReferenceDeliveryDelay = unname(parametros["aReferenceDeliveryDelay"])
##                         ,aSensOfAttractToAvailability = unname(parametros["aSensOfAttractToAvailabilit
##                         ,aSensOfAttractToPrice = unname(parametros["aSensOfAttractToPrice"])
##                         # Learning Curve Params
##                         ,aLCStrength = rep(unname(parametros["aLCStrength"]), times = N_PLAYERS)
##                         ,aInitialProductionExperience = rep(unname(parametros["aInitialProductionExper
##                         ,aRatioOfFixedToVarCost = rep(unname(parametros["aRatioOfFixedToVarCost"]), ti
##                         ,aNormalProfitMargin = rep(unname(parametros["aNormalProfitMargin"]), times = N
##                         ,aNormalCapacityUtilization = rep(unname(parametros["aNormalCapacityUtilizatio
##                         #Target Capacity Sector
##                         ,aMinimumEfficientScale = rep(unname(parametros["aMinimumEfficientScale"]), ti
##
##                         # Esta variavel é desdobrada por player.
##                         ,aWeightOnSupplyLine= rep(unname(parametros["aWeightOnSupplyLine"]), times = N_
##                         ,aTimeToPerceiveCompTargetCapacity = rep(unname(parametros["aTimeToPerceiveComp
##
##                         # Price Sector
##                         ,aPriceAdjustmentTime = unname(parametros["aPriceAdjustmentTime"])
##                         ,aSensOfPriceToCosts = rep(unname(parametros["aSensOfPriceToCosts"]), times = N
##                         ,aSensOfPriceToDSBalance = rep(unname(parametros["aSensOfPriceToDSBalance"]),
##                         ,aSensOfPriceToShare = rep(unname(parametros["aSensOfPriceToShare"]), times = N
##                         # Capacity Sector
##                         ,aSwitchForPerfectCapacity = unname(parametros["aSwitchForPerfectCapacity"])
##
##                         # Pesquisa e Desenvolvimento
##                         ,aPeDLigado = unname(parametros["aPeDLigado"])
##
##                         ,aTempoMedioRealizacaoPeD = unname(parametros["aTempoMedioRealizacaoPeD"])
##                         ,aCustoMedioPatente = unname(parametros["aCustoMedioPatente"])
##                         ,aTempoMedioAvaliacao = unname(parametros["aTempoMedioAvaliacao"])
##                         ,aTaxaRejeicao = unname(parametros["aTaxaRejeicao"])
```

```
##                       ,aTempoVencimentoPatentes = unname(parametros["aTempoVencimentoPatentes"])
##                       ,aTempodeInutilizacaoPatente = unname(parametros["aTempodeInutilizacaoPatente"]
##                       ,aPerfSlope = unname(parametros["aPerfSlope"])
##                       ,aPerfMin = unname(parametros["aPerfMin"])
##                       ,aPerfMax = unname(parametros["aPerfMax"])
##                       ,aSensOfAttractToPerformance = unname(parametros["aSensOfAttractToPerformance"]
##                       ,aReferencePerformance = unname(parametros["aReferencePerformance"])
##
##                       ,aInitialInvestimentoNaoRealizadoPeD = rep(unname(parametros["aInitialInvestime
##                       ,aInitialPatentesRequisitadas = rep(unname(parametros["aInitialPatentesRequisi
##                       ,aInitialPatentesEmpresa = rep(unname(parametros["aInitialPatentesEmpresa"]),
##                       ,aInitialsPatentesEmDominioPublicoUteis = unname(parametros["aInitialsPatentes
##                       ,aInitialsInvestimentoPeDDepreciar = rep(unname(parametros["aInitialsInvestimen
##
##                       ,aInitialReorderShare =unname(parametros["aInitialReorderShare"])
##                       ,aTotalInitialInstalledBase = unname(parametros["aTotalInitialInstalledBase"])
##                       ,aInitialIndustryShipments = unname(parametros["aInitialIndustryShipments"])
##                       ,Scenario = unname(parametros["Scenario"])
##                       ,Lever = unname(parametros["Lever"])
##       )
##
##
##       ##### VARIÁVEIS DE ENTRADA - ESTOQUES INICIAIS, SEM AJUSTES #####
##
##       # Informando Estoques Iniciais, sem ajustes, apenas para calcular o primeiro tempo.
##       stocks_iniciais  <- c(
##         sNPVProfit = rep(0, times = N_PLAYERS)
##         ,sValueOfBacklog = rep(12738001, times = N_PLAYERS)
##         ,sBacklog = rep(12738, times = N_PLAYERS)
##         ,sInstalledBase = rep(0, times = N_PLAYERS)  # rep(30000, times = N_PLAYERS) # Este estoque p
##         ,sPrice = unname(auxs$aInitialPrice)
##         ,sCumulativeAdopters = 60000 # Este estoque possui uma fórmula, verificar como fazer aqui no
##         ,sReportedIndustryVolume = rep(101904, times = N_PLAYERS)
##         ,sCumulativeProduction = rep(1e+007, times = N_PLAYERS) # Este estoque possui formula
##         ,sPerceivedCompTargetCapacity = rep(63690, times = N_PLAYERS) # Este estoque possui formula
##         ,sSmoothCapacity1 = rep(63690, times = N_PLAYERS) # Este estoque possui formula
##         ,sSmoothCapacity2 = rep(63690, times = N_PLAYERS) # Este estoque possui formula
##         ,sSmoothCapacity3 = rep(63690, times = N_PLAYERS) # Este estoque possui formula
##
##         ,sInvestimentoNaoRealizadoPeD = rep(1000, times = N_PLAYERS)
##         ,sPatentesRequisitadas = rep(1000, times = N_PLAYERS)
##         ,sPatentesEmpresa = rep(1000, times = N_PLAYERS)
##         ,sPatentesEmDominioPublicoUteis = rep(1000, times = N_PLAYERS)
##         ,sInvestimentoPeDDepreciar = rep(1000, times = N_PLAYERS)
##
##       )
##
##       # Calculando estoques para o t0.
##       iteracoes_aquecimento_estoques = 3
##
##       for(i in 1:iteracoes_aquecimento_estoques){
##
##         estoques_calculados = modelo(time = 0, stocks = stocks_iniciais, auxs = auxs, modo = "inicial
##
```

```
##         stocks_iniciais  <- c(
##           sNPVProfit = unname(stocks_iniciais[grep("sNPVProfit", x = names(stocks_iniciais))])
##           ,sValueOfBacklog = unname(estoques_calculados$ValueOfBacklogIni)
##           ,sBacklog = unname(estoques_calculados$BacklogIni)
##           ,sInstalledBase = unname(estoques_calculados$InstalledBaseIni)
##           ,sPrice = unname(stocks_iniciais[grep("sPrice", x = names(stocks_iniciais))])
##           ,sCumulativeAdopters = unname(estoques_calculados$CumulativeAdoptersIni)
##           ,sReportedIndustryVolume = rep(unname(estoques_calculados$ReportedIndustryVolumeIni), times
##           ,sCumulativeProduction = unname(estoques_calculados$CumulativeProductionIni)
##           ,sPerceivedCompTargetCapacity = unname(estoques_calculados$PerceivedCompTargetCapacityIni)
##           ,sSmoothCapacity1 = unname(estoques_calculados$CapacityIni)
##           ,sSmoothCapacity2 = unname(estoques_calculados$CapacityIni)
##           ,sSmoothCapacity3 = unname(estoques_calculados$CapacityIni)
##           ,sInvestimentoNaoRealizadoPeD = unname(estoques_calculados$InitialInvestimentoNaoRealizadoI
##           ,sPatentesRequisitadas = unname(estoques_calculados$InitialPatentesRequisitadas)
##           ,sPatentesEmpresa = unname(estoques_calculados$InitialPatentesEmpresa)
##           ,sPatentesEmDominioPublicoUteis = unname(estoques_calculados$InitialsPatentesEmDominioPubli
##           ,sInvestimentoPeDDepreciar = unname(estoques_calculados$InitialsInvestimentoPeDDepreciar)
##         )
##
##       }
##
##       stocks = stocks_iniciais
##
##       # Substituindo estoques no t0
##       # stocks  <- c(
##       #   sNPVProfit = unname(stocks_iniciais[grep("sNPVProfit", x = names(stocks_iniciais))])
##       #   ,sValueOfBacklog = unname(estoques_calculados$ValueOfBacklogIni)
##       #   ,sBacklog = unname(estoques_calculados$BacklogIni)
##       #   ,sInstalledBase = unname(estoques_calculados$InstalledBaseIni)
##       #   ,sPrice = unname(stocks_iniciais[grep("sPrice", x = names(stocks_iniciais))])
##       #   ,sCumulativeAdopters = unname(estoques_calculados$CumulativeAdoptersIni)
##       #   ,sReportedIndustryVolume = rep(unname(estoques_calculados$ReportedIndustryVolumeIni), times
##       #   ,sCumulativeProduction = unname(estoques_calculados$CumulativeProductionIni)
##       #   ,sPerceivedCompTargetCapacity = unname(estoques_calculados$PerceivedCompTargetCapacityIni)
##       #   ,sSmoothCapacity1 = unname(estoques_calculados$CapacityIni)
##       #   ,sSmoothCapacity2 = unname(estoques_calculados$CapacityIni)
##       #   ,sSmoothCapacity3 = unname(estoques_calculados$CapacityIni)
##       #   ,sInvestimentoNaoRealizadoPeD = unname(estoques_calculados$InitialInvestimentoNaoRealizadoI
##       #   ,sPatentesRequisitadas = unname(estoques_calculados$InitialPatentesRequisitadas)
##       #   ,sPatentesEmpresa = unname(estoques_calculados$InitialPatentesEmpresa)
##       #   ,sPatentesEmDominioPublicoUteis = unname(estoques_calculados$InitialsPatentesEmDominioPubli
##       #   ,sInvestimentoPeDDepreciar = unname(estoques_calculados$InitialsInvestimentoPeDDepreciar)
##       # )
##
##       resultado_completo = data.frame(deSolve::ode(y=stocks, simtime, func = modelo,
##                                           parms=auxs, method="euler"))
##       # Posso filtrar os resultados ou não:
##       # resultado_completo[variaveis_calibracao]
##       resultado_completo
##     }, modelo <- function(time, stocks, auxs, modo = "completo"){
##       with(as.list(c(stocks, auxs)),{
##         # Criando uma variavel n_tempo local
##         n_tempo = nrow(list.variaveis.globais$sReportedIndustryVolume)
```

```
##
##
##          ##### VETORIZANDO ESTOQUES #####
##          #Estoques Vetorizados = substituindo estoques pela forma vetorizada (pra que seja possivel fo
##          # Esta implementação tem por objetivo não gerar a necessidade de referenciar os estoque spelo
##          sNPVProfit = stocks[grep("sNPVProfit", x = names(stocks))]
##          sValueOfBacklog = stocks[grep("sValueOfBacklog", x = names(stocks))]
##          sBacklog = stocks[grep("sBacklog", x = names(stocks))]
##          sInstalledBase = stocks[grep("sInstalledBase", x = names(stocks))]
##          sPrice = stocks[grep("sPrice", x = names(stocks))]
##          sCumulativeAdopters = stocks[grep("sCumulativeAdopters", x = names(stocks))]
##          sReportedIndustryVolume = stocks[grep("sReportedIndustryVolume", x = names(stocks))]
##          sCumulativeProduction = stocks[grep("sCumulativeProduction", x = names(stocks))]
##          sPerceivedCompTargetCapacity = stocks[grep("sPerceivedCompTargetCapacity", x = names(stocks))
##          sSmoothCapacity1 = stocks[grep("sSmoothCapacity1", x = names(stocks))]
##          sSmoothCapacity2 = stocks[grep("sSmoothCapacity2", x = names(stocks))]
##          sSmoothCapacity3 = stocks[grep("sSmoothCapacity3", x = names(stocks))]
##
##          sInvestimentoNaoRealizadoPeD = stocks[grep("sInvestimentoNaoRealizadoPeD", x = names(stocks))
##          sPatentesRequisitadas = stocks[grep("sPatentesRequisitadas", x = names(stocks))]
##          sPatentesEmpresa = stocks[grep("sPatentesEmpresa", x = names(stocks))]
##          sPatentesEmDominioPublicoUteis = stocks[grep("sPatentesEmDominioPublicoUteis", x = names(stoc
##          sInvestimentoPeDDepreciar = stocks[grep("sInvestimentoPeDDepreciar", x = names(stocks))]
##
##          #Obtendo o número da linha no qual estou
##          linha = ((time - START)* (n_tempo - 1)) / (FINISH - START) + 1
##
##          # Gravando a Variável sReportedIndustryVolume no vetor global
##          list.variaveis.globais$sReportedIndustryVolume[linha,] <<- sReportedIndustryVolume
##
##
##          ### Calculando Variáveis para o Estoque Inicial de Cumulative Adopters
##
##          aEstimatedAdopters = aTotalInitialInstalledBase / aUnitsPerHousehold
##
##          aInitialNewAdoptersOrderRate = aInitialIndustryShipments*(1-aInitialReorderShare)
##
##          aInitialAdoptionRate = aInitialNewAdoptersOrderRate / aUnitsPerHousehold
##
##          aInitialCumulativeAdopters2 = ((aInitialAdoptionRate/(aPopulation-aEstimatedAdopters))-aInnov
##
##          aInitialCumulativeAdopters = aInitialCumulativeAdopters2
##
##
##          ##### DIFFUSION SECTOR #####
##          aDemandCurveSlope = - aReferenceIndustryDemandElasticity * (aReferencePopulation / aReferenc
##
##          aLowestPrice = min(sPrice)
##
##          aIndustryDemand = min(
##            aPopulation,
##            aReferencePopulation * max(
##              0,
##              1 + aDemandCurveSlope * (aLowestPrice - aReferencePrice) / aReferencePopulation
```

```
##           )
##         )
##
##         checkIndustryDemand = aIndustryDemand
##
##         # A fórmula abaixo não é mais utilizada.
##         # aInitialCumulativeAdopters = aInitialDiffusionFraction * aIndustryDemand
##
##         aNonAdopters = aIndustryDemand - sCumulativeAdopters
##
##         checkNonAdopters = aNonAdopters
##
##         # Ajuste temporário: Colocar o adoption Rate como Fluxo apenas positivo.
##
##         fAdoptionRate = max(0,
##                             aNonAdopters * (aInnovatorAdoptionFraction + aWOMStrength * sCumulativeA
##
##         checkAdoptionRate = fAdoptionRate
##
##         ##### ORDERS SECTOR - PT 1 #####
##
##         fDiscardRate = sInstalledBase * aFractionalDiscardRate
##
##         ##### INDUSTRY DEMAND SECTOR #####
##
##         fReorderRate = sum(fDiscardRate)
##
##         aInitialOrderRate = aUnitsPerHousehold * fAdoptionRate
##
##         fIndustryOrderRate = fReorderRate + aInitialOrderRate
##
##         checkIndustryOrderRate = fIndustryOrderRate
##
##         ##### ORDERS SECTOR - PT 2 #####
##
##         aDesiredShipments = sBacklog / aNormalDeliveryDelay
##
##         ### CAPACITY SECTOR - PT 1 ####
##
##         aCapacity = aSwitchForPerfectCapacity * (aDesiredShipments / aNormalCapacityUtilization) + (
##
##         aNormalProduction = aCapacity * aNormalCapacityUtilization
##
##         aIndustryNormalProduction = sum(aNormalProduction)
##
##         ##### ORDERS SECTOR - PT 3 #####
##
##         fShipments = aSwitchForCapacity * pmin(aDesiredShipments, aCapacity) + (1-aSwitchForCapacity
##
##         aCapacityUtilization = fShipments / aCapacity
##
##         aIndustryShipments = sum(fShipments)
##
##         aMarketShare = fShipments / aIndustryShipments
```

```
##
##         aDeliveryDelay = sBacklog / fShipments
##
##         checkIndustryShipments = aIndustryShipments
##
##         ##### MARKET SECTOR #####
##
##         # Patentes e Performance
##
##         aPatentesEmpresaTemAcesso = sPatentesRequisitadas + sPatentesEmpresa + sPatentesEmDominioPubl
##
##         aPerformanceCalculada = aPerfSlope * aPatentesEmpresaTemAcesso
##
##         aPerformance = pmax(aPerfMin, pmin(aPerfMax, aPerformanceCalculada))
##
##         checkPerformance = mean(aPerformance)
##
##         aAttractivenessFromPerformance = aPeDLigado * exp(aSensOfAttractToPerformance*(aReferencePeri
##
##         aAttractivenessFromAvailability = exp(aSensOfAttractToAvailability*(aDeliveryDelay/aReferenc
##
##         aAttractivenessFromPrice = exp(aSensOfAttractToPrice*(sPrice/aReferencePrice))
##
##         aAttractiveness = aAttractivenessFromAvailability * aAttractivenessFromPrice * aAttractivenes
##
##         aTotalAttractiveness = sum(aAttractiveness)
##
##         aOrderShare = aAttractiveness / aTotalAttractiveness
##
##         ##### ORDERS SECTOR - PT 3 #####
##
##         fOrders = fIndustryOrderRate * aOrderShare
##
##         checkOrders = sum(fOrders)
##
##         ##### EXPECTED INDUSTRY DEMAND SECTOR #####
##
##         aInitialDemandForecast = fReorderRate
##
##         aIndustryVolume = pmax(aInitialDemandForecast,
##                                aSwitchForShipmentsInForecast*aIndustryShipments+
##                                  (1-aSwitchForShipmentsInForecast)*fIndustryOrderRate)
##
##
##         # Variavel com SMOOTH - Primeira Ordem: - Retirando o DT, o calculo funcionou corretamente!
##         fsmooth_ReportedIndustryVolume = ((aIndustryVolume - sReportedIndustryVolume) / aVolumeRepor
##
##         # Variavel com DELAY - A definição das constantes aqui devem ser alteradas se as condicoes i
##         # Esta implementacao considera que os delays sempre serao iguais. Se os delays nao forem igu
##         if((time - START) > aTimeForHistoricalVolume) {
##           nlinhas_delay = aTimeForHistoricalVolume / STEP
##           aLaggedIndustryVolume = list.variaveis.globais$sReportedIndustryVolume[(linha - nlinhas_de
##         } else {
##           aLaggedIndustryVolume = list.variaveis.globais$sReportedIndustryVolume[1,]
```

```
##         }
##
##         aExpGrowthInVolume =  log(sReportedIndustryVolume/aLaggedIndustryVolume)/aTimeForHistoricalV
##
##         aExpectedIndustryDemand = sReportedIndustryVolume*exp(aForecastHorizon*aCapacityAcquisitionD
##
##         list.variaveis.globais$aExpectedIndustryDemand[linha,] <<- aExpectedIndustryDemand
##
##
##         # Mais uma variável com delay
##         if((time - START) > aCapacityAcquisitionDelay) {
##           nlinhas_delay = aCapacityAcquisitionDelay / STEP
##           aLaggedVolumeForecast = list.variaveis.globais$aExpectedIndustryDemand[linha-nlinhas_delay
##         } else {
##           aLaggedVolumeForecast = list.variaveis.globais$aExpectedIndustryDemand[1,]
##         }
##
##         aForecastError = (aLaggedVolumeForecast - aIndustryVolume)/(1e-009+aIndustryVolume)
##
##         checkLaggedVolumeForecast = mean(aLaggedVolumeForecast)
##
##         ##### TARGET CAPACITY SECTOR #####
##
##         aIndustryCapacity = sum(aCapacity)
##
##         aCompetitorCapacity = aIndustryCapacity - aCapacity
##
##         aExpectedCompCapacity = aNormalCapacityUtilization*(aWeightOnSupplyLine*sPerceivedCompTarget
##
##         aUncontestedDemand = pmax(0, aExpectedIndustryDemand - aExpectedCompCapacity)
##
##         aUncontestedMarketShare = aUncontestedDemand / aExpectedIndustryDemand
##
##         aSwitchForCapacityStrategy1 = ifelse(aSwitchForCapacityStrategy == 1, 1, 0)
##         aSwitchForCapacityStrategy2 = ifelse(aSwitchForCapacityStrategy == 2, 1, 0)
##         aSwitchForCapacityStrategy3 = ifelse(aSwitchForCapacityStrategy == 3, 1, 0)
##         aSwitchForCapacityStrategy4 = ifelse(aSwitchForCapacityStrategy == 4, 1, 0)
##
##         aTargetMarketShare = {
##           aSwitchForCapacityStrategy1*pmax(aDesiredMarketShare,aUncontestedMarketShare) +
##             aSwitchForCapacityStrategy2*pmin(aDesiredMarketShare,aUncontestedMarketShare) +
##             aSwitchForCapacityStrategy3*aDesiredMarketShare +
##             aSwitchForCapacityStrategy4*aUncontestedMarketShare
##         }
##
##
##         aTargetCapacity = pmax(aMinimumEfficientScale,
##                                aTargetMarketShare*aExpectedIndustryDemand/aNormalCapacityUtilization
##
##         aTargetNormalProduction = aTargetCapacity * aNormalCapacityUtilization
##
##         aIndustryTotalTargetCapacity = sum(aTargetCapacity)
##
##         aCompetitorTargetCapacity = aIndustryTotalTargetCapacity - aTargetCapacity
```

```
##
##         fChangePerceivedCompTargetCapacity = (aCompetitorTargetCapacity - sPerceivedCompTargetCapaci
##
##         checkCompetitorTargetCapacity = mean(aCompetitorTargetCapacity)
##
##         ##### CAPACITY SECTOR  - PT 2 - FLUXOS #####
##         fchangeSmoothCapacity1 = (aTargetCapacity - sSmoothCapacity1) / (aCapacityAcquisitionDelay /
##         fchangeSmoothCapacity2 = (sSmoothCapacity1 - sSmoothCapacity2) / (aCapacityAcquisitionDelay /
##         fchangeSmoothCapacity3 = (sSmoothCapacity2 - sSmoothCapacity3) / (aCapacityAcquisitionDelay /
##
##
##
##         ##### Custo P e D ####
##         aTempoDepreciacao = aTempoMedioAvaliacao + aTempoVencimentoPatentes + aTempoMedioRealizacaoP
##
##         fDepreciacaoInvPeD = sInvestimentoPeDDepreciar / aTempoDepreciacao
##
##         aPeDUnitCost = fDepreciacaoInvPeD / fShipments
##
##
##
##
##         ##### LEARNING CURVE SECTOR #####
##         fProduction = fShipments
##
##         aLCExponent = log(aLCStrength)/log(2)
##
##         aLearning = (sCumulativeProduction/aInitialProductionExperience)^aLCExponent
##
##         aInitialUnitFixedCost = (aInitialPrice/(1+aNormalProfitMargin))*aRatioOfFixedToVarCost*(1/(1
##
##         aInitialUnitVariableCost = (aInitialPrice/(1+aNormalProfitMargin))*(1/(1+aRatioOfFixedToVarC
##
##         aUnitFixedCost = aLearning * aInitialUnitFixedCost + aPeDUnitCost * aPeDLigado
##
##         aUnitVariableCost = aLearning * aInitialUnitVariableCost
##
##         checkUnitFixedCost = mean(aUnitFixedCost)
##
##         checkUnitVariableCost = mean(aUnitVariableCost)
##
##         ##### PRICE SECTOR #####
##
##         aBasePrice = (1+aNormalProfitMargin)*(aUnitVariableCost+aUnitFixedCost/aNormalCapacityUtiliz
##
##         aDemandSupplyBalance = aDesiredShipments/(aNormalCapacityUtilization*aCapacity)
##
##         aTargetPrice =
##           pmax(aUnitVariableCost,
##                sPrice*
##                  (1+aSensOfPriceToCosts*((aBasePrice/sPrice)-1))*
##                  (1+aSensOfPriceToDSBalance*(aDemandSupplyBalance-1))*
##                  (1+aSensOfPriceToShare*((aTargetMarketShare-aMarketShare))))
##
```

```
##         checkTargetPrice = mean(aTargetPrice)
##
##         fChangeInPrice = (aTargetPrice - sPrice) / aPriceAdjustmentTime
##
##         ##### NET INCOME SECTOR #####
##
##         aDiscountFactor = exp(-aDiscountRate*(time - START)) #
##
##         fValueOfNewOrders = fOrders * sPrice
##
##         checkValueOfNewOrders1 = fValueOfNewOrders[1] #
##
##         aAveragePriceOfOrderBook = sValueOfBacklog / sBacklog
##
##         fRevenue = fShipments * aAveragePriceOfOrderBook #
##
##         ##### P&D - Investimento #####
##
##         fInvestimentoPeD = fRevenue * aOrcamentoPeD * aPeDLigado
##
##         fInvestimentoPeDRealizado = sInvestimentoNaoRealizadoPeD / aTempoMedioRealizacaoPeD
##
##         fPatentesSolicitadas = ((1-aPercPeDAberto) * fInvestimentoPeDRealizado) / aCustoMedioPatente
##
##         fPatentesRejeitadas = (sPatentesRequisitadas/aTempoMedioAvaliacao) * aTaxaRejeicao
##
##         fPatentesConcedidas = (sPatentesRequisitadas/aTempoMedioAvaliacao) * (1-aTaxaRejeicao)
##
##         # Estou somando as Patentes com investimenti (direto em domínio publico na equação abaixo, s
##         # Eventualmente é possível modelar este comportamento passando por outros estoques.
##         fPatentesVencidas = sPatentesEmpresa / aTempoVencimentoPatentes
##
##         fPatentesAbertas = ((aPercPeDAberto * fInvestimentoPeDRealizado) / aCustoMedioPatente) * (1-a
##
##         fPatentesUtilidadeExpirada = sPatentesEmDominioPublicoUteis / aTempodeInutilizacaoPatente
##
##         ##### NET INCOME - PARTE 2 #####
##
##         checkRevenue1 = fRevenue[1] #
##
##         aVariableCost = fShipments * aUnitVariableCost #
##
##         aFixedCost = aCapacity * (aUnitFixedCost - (aPeDUnitCost * aPeDLigado)) #
##
##         fCost = aFixedCost + aVariableCost #
##
##         fNetIncome = fRevenue - fCost - fInvestimentoPeD #
##
##         fNPVProfitChange = fNetIncome * aDiscountFactor #
##
##         checkNPVProfitChange = mean(fNPVProfitChange) #
##
##         checkNPVProfitChange1 = fNPVProfitChange[1]
##
```

```
##         aNPVIndustryProfits = sum(sNPVProfit) #
##
##
##
##         ##### ESTOQUES #####
##
##         d_NPVProfit_dt = fNPVProfitChange
##
##         d_ValueOfBacklog_dt = fValueOfNewOrders - fRevenue
##
##         d_Backlog_dt = fOrders - fShipments
##
##         d_InstalledBase_dt = fShipments - fDiscardRate
##
##         d_Price_dt = fChangeInPrice
##
##         d_CumulativeAdopters_dt = fAdoptionRate
##
##         d_sReportedIndustryVolume_dt = fsmooth_ReportedIndustryVolume
##
##         d_CumulativeProduction_dt = fProduction
##
##         d_PerceivedCompTargetCapacity_dt = fChangePerceivedCompTargetCapacity
##
##         d_SmoothCapacity1_dt = fchangeSmoothCapacity1
##
##         d_SmoothCapacity2_dt = fchangeSmoothCapacity2
##
##         d_SmoothCapacity3_dt = fchangeSmoothCapacity3
##
##         #Estoques do Investimento em PeD
##
##         d_InvestimentoNaoRealizadoPeD_dt = fInvestimentoPeD - fInvestimentoPeDRealizado
##
##         d_PatentesRequisitadas_dt = fPatentesSolicitadas - fPatentesConcedidas - fPatentesRejeitadas
##
##         d_PatentesEmpresa_dt = fPatentesConcedidas - fPatentesVencidas
##
##         d_PatentesEmDominioPublicoUteis_dt = sum(fPatentesVencidas) - fPatentesUtilidadeExpirada
##
##         d_InvestimentoPeDDepreciar_dt = fInvestimentoPeD - fDepreciacaoInvPeD
##
##
##
##         # Variaveis de Estoques Iniciais
##
##         BacklogIni = aInitialSharePlayers * fIndustryOrderRate * aNormalDeliveryDelay
##
##         InstalledBaseIni = aInitialCumulativeAdopters * aInitialSharePlayers * aUnitsPerHousehold
##
##         CumulativeAdoptersIni = aInitialCumulativeAdopters
##
##         ValueOfBacklogIni = aInitialSharePlayers * fIndustryOrderRate * aNormalDeliveryDelay * aInit
##
```

```
##         ReportedIndustryVolumeIni = aIndustryVolume
##
##         CumulativeProductionIni = aInitialProductionExperience
##
##         PerceivedCompTargetCapacityIni = aCompetitorCapacity
##
##         CapacityIni = aInitialSharePlayers * fIndustryOrderRate / aNormalCapacityUtilization
##
##         InitialInvestimentoNaoRealizadoPeD = aInitialInvestimentoNaoRealizadoPeD * aPatentShare
##
##         InitialPatentesRequisitadas = aInitialPatentesRequisitadas * aPatentShare
##
##         InitialPatentesEmpresa = aInitialPatentesEmpresa * aPatentShare
##
##         InitialsPatentesEmDominioPublicoUteis =  aInitialsPatentesEmDominioPublicoUteis
##
##         InitialsInvestimentoPeDDepreciar = aInitialsInvestimentoPeDDepreciar * aPatentShare
##
##
##         ##### ESTOQUES - INICIAIS #####
##
##         stocks_ini = list(
##           BacklogIni = BacklogIni,
##           InstalledBaseIni = InstalledBaseIni,
##           CumulativeAdoptersIni = CumulativeAdoptersIni,
##           ValueOfBacklogIni = ValueOfBacklogIni,
##           ReportedIndustryVolumeIni = ReportedIndustryVolumeIni,
##           CumulativeProductionIni = CumulativeProductionIni,
##           PerceivedCompTargetCapacityIni = PerceivedCompTargetCapacityIni,
##           CapacityIni = CapacityIni,
##
##           InitialInvestimentoNaoRealizadoPeD = InitialInvestimentoNaoRealizadoPeD,
##           InitialPatentesRequisitadas = InitialPatentesRequisitadas,
##           InitialPatentesEmpresa = InitialPatentesEmpresa,
##           InitialsPatentesEmDominioPublicoUteis = InitialsPatentesEmDominioPublicoUteis,
##           InitialsInvestimentoPeDDepreciar = InitialsInvestimentoPeDDepreciar
##         )
##
##         ##### COMPARAR RESULTADOS COM O ITHINK #####
##
##         if(VERIFICAR_STOCKS & modo == "completo"){
##           for (variavel in variaveis_ithink_stocks) {
##             # Definir o tipo de variavel
##             # Variavel é um estoque?
##             variavel_ithink_alterada = gsub(pattern = "\\[", replacement = "", x = variavel, ignore.
##             variavel_ithink_alterada = gsub(pattern = "\\]", replacement = "", x = variavel_ithink_al
##
##             # Verificar apenas Estoques:
##             variavel_ithink_alterada = paste("s", variavel_ithink_alterada, sep = "")
##
##             # Valor da Variavel Calculada
##             valor_variavel_R = eval(parse(text = variavel_ithink_alterada))
##
##             valor_variavel_ithink = dados_ithink_stocks[[linha,variavel]]
```

```
##
##             diferenca = valor_variavel_R - valor_variavel_ithink
##
##             if (abs(x = diferenca) > CHECK_PRECISION){
##               message(paste("Estoque Diff:", time, linha, variavel, diferenca, sep = " - "))
##               if(BROWSE_ON_DIFF){
##                 browser()
##               }
##             }
##           }
##         }
##
##
##         if(VERIFICAR_CHECKS & modo == "completo"){
##           for (variavel in variaveis_ithink_checks) {
##             # Definir o tipo de variavel
##             # Variavel é um estoque?
##             variavel_ithink_alterada = gsub(pattern = "\\[", replacement = "", x = variavel, ignore.
##             variavel_ithink_alterada = gsub(pattern = "\\]", replacement = "", x = variavel_ithink_al
##
##             # Verificar apenas Estoques:
##             #variavel_ithink_alterada = paste("s", variavel_ithink_alterada, sep = "")
##
##             # Valor da Variavel Calculada
##             valor_variavel_R = eval(parse(text = variavel_ithink_alterada))
##
##             valor_variavel_ithink = dados_ithink_checks[[linha,variavel]]
##
##             diferenca = valor_variavel_R - valor_variavel_ithink
##
##             if(!is.na(diferenca)){
##               if (abs(x = diferenca) > CHECK_PRECISION){
##                 message(paste("Check Diff:", time, linha, variavel, diferenca, sep = " - "))
##                 if(BROWSE_ON_DIFF){
##                   browser()
##                 }
##               }
##             }
##
##           }
##         }
##
##         # Colocar isso dentro do IF abaixo e verificar!
##         if(VERIFICAR_GLOBAL & modo == "completo"){
##
##           # Forma que usa todas as variaveis do ambiente:
##           # variaveis_disponiveis_ambiente = ls()
##           # variaveis_auxiliares = variaveis_disponiveis_ambiente[grep("^[aA].*", variaveis_disponiv
##           #
##           # Forma que usa as variaveis globais definidas em um vetor
##           variaveis_auxiliares = variaveis_globais_a_verificar
##
##
##           seletor_players = paste("[",1:N_PLAYERS,"]", sep = "")
```

```
##
##           variaveis_a_verificar = expand.grid(variaveis_auxiliares, seletor_players)
##
##           variaveis_a_verificar = paste(variaveis_a_verificar[,1], variaveis_a_verificar[,2], sep = "
##
##           variaveis_a_verificar_no_ithink = substring(variaveis_a_verificar, 2)
##
##
##           verificar_variaveis_globais = function(n_variavel){
##             valor_variavel_R = eval(parse(text = variaveis_a_verificar[n_variavel]))
##
##             valor_variavel_ithink = dados_ithink_global[[linha,variaveis_a_verificar_no_ithink[n_vari
##
##             if((length(valor_variavel_ithink) > 0)) {
##               decisao = !is.na(valor_variavel_ithink) & is.numeric(valor_variavel_ithink)
##               if(decisao == FALSE) {
##                 valor_variavel_ithink = NA
##               }
##             } else {valor_variavel_ithink = NA}
##
##
##             if((length(valor_variavel_R) > 0)) {
##               decisao = !is.na(valor_variavel_R) & is.numeric(valor_variavel_R)
##               if(decisao == FALSE) {
##                 valor_variavel_R = NA
##               }
##             } else {valor_variavel_R = NA}
##
##
##             diferenca = unname(valor_variavel_R)  - unname(valor_variavel_ithink)
##
##             diferenca
##           }
##
##           diferencas = lapply(X = 1:length(variaveis_a_verificar), FUN = verificar_variaveis_globais)
##
##           diferencas = do.call(rbind, diferencas)
##
##           matriz_diferencas = data.frame(
##             variaveis_a_verificar,
##             diferencas
##           )
##
##           diferencas_a_reportar = subset(matriz_diferencas, abs(diferencas) > CHECK_PRECISION)
##
##           if(nrow(diferencas_a_reportar)>1) {
##             message(paste("Check Diferenças Globais:", time, linha, sep = " - "))
##             if(BROWSE_ON_DIFF){
##               browser()
##             }
##           }
##
##         }
##
```

```
##         # if(modo == "completo" & time == FINISH) {
##         #   browser()
##         # }
##         ##### VARIÁVEIS RETORNADAS #####
##
##         resultado_completo = list(c(
##           d_NPVProfit_dt
##           ,d_ValueOfBacklog_dt
##           ,d_Backlog_dt
##           ,d_InstalledBase_dt
##           ,d_Price_dt
##           ,d_CumulativeAdopters_dt
##           ,d_sReportedIndustryVolume_dt
##           ,d_CumulativeProduction_dt
##           ,d_PerceivedCompTargetCapacity_dt
##           ,d_SmoothCapacity1_dt
##           ,d_SmoothCapacity2_dt
##           ,d_SmoothCapacity3_dt
##           ,d_InvestimentoNaoRealizadoPeD_dt
##           ,d_PatentesRequisitadas_dt
##           ,d_PatentesEmpresa_dt
##           ,d_PatentesEmDominioPublicoUteis_dt
##           ,d_InvestimentoPeDDepreciar_dt
##         )
##         ,fIndustryOrderRate = unname(fIndustryOrderRate)
##         ,aOrderShare = unname(aOrderShare)
##         ,aPerformance = unname(aPerformance)
##         ,aNonAdopters = unname(aNonAdopters)
##         ,fReorderRate = unname(fReorderRate)
##         ,aIndustryShipments = unname(aIndustryShipments)
##         ,aIndustryVolume = unname(aIndustryVolume)
##         ,fNPVProfitChange = unname(fNPVProfitChange)
##         ,fNetIncome = unname(fNetIncome)
##         ,aNPVIndustryProfits = unname(aNPVIndustryProfits))
##
##         return (if(modo == "inicial"){
##           stocks_ini
##         } else {
##           resultado_completo
##         })
##       })
##     }, nomes_variaveis = c("Tempo", "d_NPVProfit_dt", "aDiscountFactor", "aDiscountRate", "fNPVProfi
##     sdmodel = list(
##       Start = START,
##       Finish = FINISH,
##       Step = STEP,
##       SimTime = SIM_TIME,
##       # Auxs = auxs,
##       # Stocks = stocks,
##       Modelo = modelo,
##       Variaveis = nomes_variaveis
##     ), simularRDM_e_escolher_estrategia = function(inputs = "params.xlsx", sdmodel = sdmodel, opcoes
##
##       output_simulacao = simular_RDM(arquivo_de_inputs=inputs ,sdmodel = sdmodel, n = opcoes$N, opco
```

```
##
##       ## Simular
##       dados_simulacao = output_simulacao$DadosSimulacao
##
##       # Selecionando dados do último ano:
##       dados = selecionar_ultimo_periodo(dados_simulacao = dados_simulacao, var_tempo = opcoes$VarTemp
##
##       # Analisar Regret
##       analise_regret = calcular_e_resumir_regret(dados = dados, var_resposta = opcoes$VarResposta, va
##
##       # Escolher a Estratégia Candidata, com base no critério de robustez dos percentis
##       estrategia_candidata = escolher_estrategia_candidata(dados = analise_regret$Dados, resumo_estra
##
##       message(paste("A Estrategia candidata é a ", estrategia_candidata$Lever))
##
##       output = list(
##         DadosSimulados = dados_simulacao,
##         DadosUltimoPeriodo = dados,
##         AnaliseRegret = analise_regret,
##         Inputs = output_simulacao$Inputs,
##         Ensemble = output_simulacao$Ensemble,
##         EstrategiaCandidata =  as.numeric(estrategia_candidata[opcoes$VarEstrategias]),
##         Opcoes = opcoes,
##         SdModel = sdmodel
##       )
##
##       output
##
##     }, carregar_inputs = function (arquivo_de_inputs="params.xlsx", abas_a_ler = c("params", "levers
##
##       # Criando uma list para os inputs
##       message(
##         paste("01. funcoes.R/carregar_inputs: Iniciando Carregamento de Inputs (funcao carregar_input
##               "arquivo_de_inputs = ", arquivo_de_inputs)
##       )
##       inputs = vector(mode = "list", length = length(nomes_inputs))
##       names(inputs) = nomes_inputs
##
##       # Preenchendo os Dados dos Inputs
##       for (aba in abas_a_ler) {
##         n_aba = which(aba == abas_a_ler)
##         inputs[[n_aba]] = readxl::read_excel(arquivo_de_inputs,sheet = aba)
##       }
##
##       message("01. funcoes.R/carregar_inputs: Finalizando Carregamento de Inputs.")
##       return(inputs)
##
##     }, obter_lhs_ensemble = function (params, n=100, opcoes = opcoes) {
##       message("01. funcoes.R/obter_lhs_ensemble: Iniciando Obtenção do Ensemble.")
##       #Obtendo DataFrame de Parâmetros
##
##       nvar = length(params$Variavel)
##       pontos = n
##
```

```
##       # Obtendo um Hypercubo com as Variáveis que eu quero
##       randomLHS <- lhs::randomLHS(pontos, nvar)
##
##       p = as.data.frame(randomLHS)
##       min = as.vector(params$Min)
##       max = as.vector(params$Max)
##       variaveis = as.vector(params$Variavel)
##
##       # Transformando o Hypercubo em variáveis
##       # var <- matrix(nrow=pontos, ncol=variaveis)
##       ensemble = matrix(nrow = pontos, ncol = nvar+1)
##
##       # Montando o Ensemble
##       for (var in variaveis) {
##         i = which(x = variaveis == var)
##
##         # Aqui o i é +1 porque a primeira coluna será o cenário.
##         ensemble[,i+1] = qunif(p = randomLHS[,i], min = min[i], max = max[i])
##       }
##
##       # Adicionando A variável "Scenario"
##       variaveis = c(c(opcoes$VarCenarios),variaveis)
##
##       colnames(ensemble) = variaveis
##
##       ensemble[,opcoes$VarCenarios] = 1:nrow(ensemble)
##
##       ensemble
##     }, ampliar_ensemble_com_levers = function(ensemble, levers, levers_full, opcoes) {
##
##       variaveis_adicionais = names(dplyr::select(levers, -LeverCode, -CasoBase))
##
##         # Filtrar cenários a simular caso seja necessário apenas simular o Caso Base:
##       if(opcoes$SimularApenasCasoBase){
##         levers = subset(levers, as.logical(CasoBase))
##       }
##
##       linhas_ensemble_incial = nrow(ensemble)
##       novo_ensemble = matrix(0, nrow = nrow(ensemble)*length(levers$Lever), ncol = ncol(ensemble) + l
##
##       names_old_ensemble = colnames(ensemble)
##       names_novo_ensemble = c(names_old_ensemble, variaveis_adicionais)
##
##       colnames(novo_ensemble) = names_novo_ensemble
##
##       j = 1
##       for (l in seq_along(levers$Lever)) {
##         lini = j
##         lfim = j + linhas_ensemble_incial-1
##         matriz_var_adicionais = as.matrix(levers[l,variaveis_adicionais])
##         novo_ensemble[lini:lfim,names_old_ensemble] = ensemble
##         novo_ensemble[lini:lfim,variaveis_adicionais] = matrix(matriz_var_adicionais, nrow = linhas_
##         j = j + linhas_ensemble_incial
##       }
```

```
##
##       novo_ensemble
##
##     }, simular = function(simtime, modelo, ensemble, nomes_variaveis_final, opcoes = opcoes) {
##       message("01. funcoes.R/simular: Iniciando Simulação.")
##
##       paralelo = opcoes$Paralelo
##
##       modo_paralelo = opcoes$ModoParalelo
##
##       # Rodando a Simulação (uma vez), com a primeira linha do ensemble - Ajuda a saber se funciona.
##       # Esta função apenas funciona com o estoque inicial fixo, será necessário implementar de outra
##       t_inicio_teste = Sys.time()
##       o = as.data.frame(solve_modelo_dissertacao(parametros = ensemble[1,], modelo = modelo, simtime
##
##       t_fim_teste = Sys.time()
##       t_uma_rodada = as.numeric(difftime(time1 = t_fim_teste, time2 = t_inicio_teste, units = "secs"
##
##       solve_modelo = function(n_linha_ensemble) {
##         params = ensemble[n_linha_ensemble,]
##         res = solve_modelo_dissertacao(parametros = params, modelo = modelo, simtime = simtime)
##         # Gerar Matriz de Resultados
##         cbind(res,
##               Lever = ensemble[n_linha_ensemble,opcoes$VarEstrategias],
##               Scenario = ensemble[n_linha_ensemble,opcoes$VarCenarios])
##       }
##
##       if(paralelo == TRUE) {
##
##         t_inicio = Sys.time()
##         message(t_inicio)
##
##         if((modo_paralelo == "PSOCK") | (modo_paralelo == "FORK")){
##           # Calculate the number of cores
##           no_cores <- detectCores() - 1
##           # Inicializar Cluster
##           cl <- makeCluster(no_cores, type = modo_paralelo)
##
##           # Carregando bibliotecas no cluster:
##           #clusterEvalQ(cl, source("funcoes.R"))
##           #clusterEvalQ(cl, eval(parse('funcoes.R')))
##           # textConnection(animal.R)
##
##           if(modo_paralelo  == "PSOCK") {
##             # Se o modo paralelo é "PSOCK", é necessário definir explícitamente as variáveis que dev
##             # http://gforge.se/2015/02/how-to-go-parallel-in-r-basics-tips/
##             clusterEvalQ(cl, library(deSolve))
##
##             # Exportando objetos que preciso ter nos clusters:
##             clusterExport(cl, varlist = list("ensemble",
##                                              "modelo",
##                                              "simtime",
##                                              "FINISH",
##                                              "VERIFICAR_STOCKS",
```

```
##                                             "solve_modelo",
##                                             "VERIFICAR_CHECKS",
##                                             "VAR_LEVER",
##                                             "VAR_SCENARIO",
##                                             "opcoes",
##                                             "STEP",
##                                             "solve_modelo_dissertacao"), envir = environment())
##           }
##
##
##           message(paste("Iniciando Simulacao em modo paralelo. Usando", no_cores, "núcleos."))
##
##           # t_uma_rodada = 0.583717
##
##           t_estimado = t_uma_rodada * nrow(ensemble)/no_cores
##           message(paste("Tempo estimado (segundos):"),t_estimado)
##
##           # Aplicando Função paralela para a computação dos resultados
##           dados_simulacao <- parLapply(cl, 1:nrow(ensemble), solve_modelo)
##           stopCluster(cl)
##
##
##         }
##
##         if(modo_paralelo == "Azure") {
##           # 3. Set your credentials - you need to give the R session your credentials to interact wi
##           setCredentials("credentials.json")
##
##           # 4. Register the pool. This will create a new pool if your pool hasn't already been provis
##           cluster <- makeCluster("cluster.json")
##
##           # 5. Register the pool as your parallel backend
##           registerDoAzureParallel(cluster)
##
##           # 6. Check that your parallel backend has been registered
##           getDoParWorkers()
##
##           # GErando resultado
##           # browser()
##           dados_simulacao <- foreach(i = 1:nrow(ensemble)) %dopar% {
##             # This code is executed, in parallel, across your cluster.
##             solve_modelo()
##           }
##
##         }
##
##         # Unindo Dados da Simulação, seja entregues pelo Azure seja pelo meu
##         dados_simulacao = do.call(rbind, dados_simulacao)
##
##         t_fim = Sys.time()
##
##         perf = t_estimado / as.numeric(difftime(time1 = t_fim, time2 = t_inicio, units = "secs")) / (
##         message(t_fim)
##         message("Finalizando Simulacao. Finalizando Cluster")
```

```
##         message(paste("Indice de Performance",perf))
##         #resultados_paralelo = lapply(1:nrow(ensemble), solve_modelo)
##         dados_simulacao = as.data.frame(dados_simulacao)
##
##       }
##
##       if(paralelo == FALSE){
##         # J é o índice dos dados simulados
##         j = 1
##         # Rodando a Simulacao Em todo o Ensemble
##         # Fazer os calculos da maneira anterior aqui.
##         nomes_temporario = names(o)
##
##         # o<-data.frame(ode(y=stocks, times=simtime, func = modelo,
##         #                   parms=ensemble[1,], method="euler"))
##         pontos = nrow(ensemble)
##
##         nlinhas = nrow(o)
##
##         ncolunas = ncol(o)+2
##
##         # Montando uma matriz com todos os dados para a simulação
##         dados_simulacao = matrix(nrow = pontos*nlinhas, ncol = ncolunas)
##
##         for (i in 1:nrow(ensemble)) {
##           resultados_simulacao = as.data.frame(solve_modelo_dissertacao(parametros = ensemble[i,], m
##
##           resultados_simulacao = as.matrix(resultados_simulacao)
##
##           linhas = nrow(resultados_simulacao)
##
##           # Avançando a linha inicial e Final da Simulação
##           l_inicial = j
##           l_final = j + linhas-1
##
##           # Adicionando o resultado ao ensemble
##           dados_simulacao[l_inicial:l_final,1:(ncolunas-2)] = resultados_simulacao
##
##           # Adicionando o Número do Lever
##           dados_simulacao[l_inicial:l_final,(ncolunas-1)] = ensemble[i,opcoes$VarEstrategias]
##
##           # Adicionando o Número do Cenário
##           dados_simulacao[l_inicial:l_final,ncolunas] = ensemble[i,opcoes$VarCenarios]
##
##           # Exibindo uma Mensagem de Status
##           if (i %% 5 == 0) {
##             message(paste(i, "simulações finalizadas."))
##           }
##           # Avançando o índice dos dados simulados
##           j = j + linhas
##         }
##         # Usando nomes temporario
##         colnames(dados_simulacao) = c(nomes_temporario, opcoes$VarEstrategias, opcoes$VarCenarios)
##         # colnames(dados_simulacao) = nomes_variaveis_final
```

```
##
##         dados_simulacao = as.data.frame(dados_simulacao)
##         names(dados_simulacao) = c(nomes_temporario, opcoes$VarEstrategias, opcoes$VarCenarios)
##         #names(dados_simulacao) = nomes_variaveis_final
##       }
##
##       message("01. funcoes.R/simular: Finalizando Simulacao.")
##
##       dados_simulacao
##     }, simular_RDM = function(arquivo_de_inputs="params.xlsx", sdmodel, n = opcoes$N, opcoes = opcoes
##       t_inicio = Sys.time()
##       message("Bem vindo ao SIMULADOR RDM! Pedro Lima.")
##       message(paste("Iniciando Simulacao RDM: ", t_inicio))
##
##       # Carregando Inputs
##       inputs = carregar_inputs(arquivo_de_inputs = arquivo_de_inputs)
##
##       # Substituindo Levers por Proketo Fatorial Completo, se isto foi selecionado:
##       # Gerar um Fatorial Completo das Variáveis, se for necessário
##       if(opcoes$FullFactorialDesign){
##         var_levers = na.omit(expand.grid(inputs$LeversFull))
##         n_levers = nrow(var_levers)
##         inputs$Levers = data.frame(Lever = 1:n_levers,
##                                    LeverCode = as.character(1:n_levers),
##                                    CasoBase = c(1, rep(0, n_levers-1)),
##                                    var_levers)
##       }
##
##
##       # Obter Ensemble LHS (Sem Variáveis das Estratégias)
##
##       # Se um ensemble não foi informado, gerar um ensemble.
##       if(missing(ensemble)){
##         # Usar o primeiro lever que eu achar
##         ensemble = obter_lhs_ensemble(params = inputs$Parametros, n = n, opcoes = opcoes)
##       }
##
##       # Ampliar Ensemble com as variáveis das Estratégias
##       novo_ensemble = ampliar_ensemble_com_levers(ensemble = ensemble, levers = inputs$Levers,levers_
##
##       # Rodando a Simulação
##       nestrategias = length(inputs$Levers$Lever)
##       nfuturos = nrow(ensemble)
##       ntempo = ((sdmodel$Finish - sdmodel$Start)/sdmodel$Step)
##
##       message(paste("Esta rotina realizará", nestrategias * nfuturos, "Simulacoes.\n (", nestrategias
##
##       # TODO: Esta Chamada vai precisar mudar para considerar a nova funcao
##       dados_simulacao = simular(simtime = sdmodel$SimTime, modelo = sdmodel$Modelo, ensemble = novo_e
##
##       t_fim = Sys.time()
##
##       message("Finalizando Simulacao. Tempo de Simulacao: ", t_fim - t_inicio)
##
```

```
##       output = list(
##         Inputs = inputs,
##         Ensemble = ensemble,
##         NovoEnsemble = novo_ensemble,
##         DadosSimulacao = dados_simulacao
##       )
##
##       output
##
##     }, calcular_regret = function(dados, var_resposta, var_group) {
##       var_maximo = paste("MaximoPor", var_group, sep = "")
##       var_minimo = paste("MinimoPor", var_group, sep = "")
##       var_regret = paste(var_resposta, "Regret", sep = "")
##       var_regret_perc = paste(var_regret, "Perc", sep = "")
##
##       dados[var_maximo] = calcular_maximo_por_variavel(var_resposta = var_resposta, var_group = var_
##
##       dados[var_minimo] = calcular_minimo_por_variavel(var_resposta = var_resposta, var_group = var_
##
##       dados[var_regret] = dados[var_maximo] - dados[var_resposta]
##
##       dados[var_regret_perc] = dados[var_regret] / (dados[var_maximo] - dados[var_minimo])
##
##       dados
##     }, resumir_variavel_resposta = function(dados = dados_ano_final, var_resposta = "Cash", var_grou
##       var_regret = paste(var_resposta, "Regret", sep = "")
##       var_regret_perc = paste(var_regret, "Perc", sep = "")
##
##       call = substitute(
##         expr =
##           dplyr::group_by(dados, VarGroup)
##         %>% dplyr::select(VarGroup, VarResposta, VarRegret, VarRegretPerc)
##         %>% dplyr::summarise(VarMedio = mean(VarResposta, na.rm = TRUE),
##                              VarDev = sd(VarResposta, na.rm = TRUE),
##                              Percentil25Var = quantile(VarResposta, probs = c(0.25), na.rm = TRUE),
##                              Percentil75Var = quantile(VarResposta, probs = c(0.75), na.rm = TRUE),
##                              RegretMedio = mean(VarRegret, na.rm = TRUE),
##                              DesvioRegret = sd(VarRegret, na.rm = TRUE),
##                              Percentil25Regret = quantile(VarRegret, probs = c(0.25), na.rm = TRUE),
##                              Percentil75Regret = quantile(VarRegret, probs = c(0.75), na.rm = TRUE),
##                              RegretMedioPerc = mean(VarRegretPerc, na.rm = TRUE),
##                              DesvioRegretPerc = sd(VarRegretPerc, na.rm = TRUE),
##                              Percentil25RegretPerc = quantile(VarRegretPerc, probs = c(0.25), na.rm = TRUE)
##                              Percentil75RegretPerc = quantile(VarRegretPerc, probs = c(0.75), na.rm = TRUE)
##         )
##         ,
##         env = list(VarGroup = as.name(var_group),
##                    VarResposta = as.name(var_resposta),
##                    VarRegret = as.name(var_regret),
##                    VarRegretPerc = as.name(var_regret_perc)
##         )
##       )
##
##       resumo = eval(call)
```

```
##
##       colnames(resumo) = c(
##         var_group,
##         paste(var_resposta, "Medio", sep = ""),
##         paste(var_resposta, "Desvio", sep = ""),
##         paste(var_resposta, "Percentil25", sep = ""),
##         paste(var_resposta, "Percentil75", sep = ""),
##         paste(var_regret, "Medio", sep = ""),
##         paste(var_regret, "Desvio", sep = ""),
##         paste(var_regret, "Percentil25", sep = ""),
##         paste(var_regret, "Percentil75", sep = ""),
##         paste(var_regret_perc, "Medio", sep = ""),
##         paste(var_regret_perc, "Desvio", sep = ""),
##         paste(var_regret_perc, "Percentil25", sep = ""),
##         paste(var_regret_perc, "Percentil75", sep = "")
##       )
##
##       resumo
##     }, escolher_estrategia_candidata = function(dados, resumo_estrategias, var_resposta, var_criterio
##
##       var_respota_criterio = paste(var_resposta, var_criterio, sep = "")
##
##
##       # Esta lista de criterios deve ser mantida igual à lista que a funcao resumir_variavel_resposta
##       possiveis_var_criterios = c("Percentil25", "Percentil75", "Medio", "Desvio", "RegretMedio", "Re
##
##       # Conferindo alguns pressupostos basicos:
##       possiveis_var_respota_e_criterios = paste(var_resposta, possiveis_var_criterios, sep = "")
##
##       # Conferindo se a variável de resposta e variável de critério combinam corretamente:
##       if (!all(possiveis_var_respota_e_criterios %in% names(resumo_estrategias))){
##         stop("Existe algo errado com a sua variavel de resposta ou variavel de criterio (a combinacao
##       }
##
##       # Conferindo se a Variavel de criterio está correta.
##       if(!var_criterio %in% possiveis_var_criterios){
##         stop(paste("Esta variavel de criterio esta incorreta. escolha entre:",possiveis_var_criterios
##       }
##
##
##       # Agora sim, posso escolhenr a estratégia que tem o menor percentil percentual 75 (assim como l
##       estrategias_candidatas = switch(sentido,
##                                       "min" = escolher_estrategia_min(resumo_estrategias, var_respota
##                                       "max" = escolher_estrategia_max(resumo_estrategias, var_respota
##
##       estrategias_candidatas
##     }, calcular_e_resumir_regret = function(dados, var_resposta, var_cenarios, var_estrategias) {
##       dados = calcular_regret(dados = dados, var_resposta = var_resposta, var_group = var_cenarios)
##
##       # Resumindo Variável de Resposta Cash:
##       resumo_estrategias = resumir_variavel_resposta(dados = dados, var_resposta = var_resposta, var_
##
##       # Formar lista de outputs dessta análise
##       output = list(
```

```
##         Dados = dados,
##         ResumoEstrategias = resumo_estrategias
##       )
##
##       output
##     }, escolher_estrategia_min = function(resumo_estrategias, criterio) {
##       linha_estrategia = which(resumo_estrategias[criterio] == min(resumo_estrategias[criterio]))
##       estrategia = resumo_estrategias[linha_estrategia, "Lever"]
##       estrategia
##     }, escolher_estrategia_max = function(resumo_estrategias, criterio) {
##       linha_estrategia = which(resumo_estrategias[criterio] == max(resumo_estrategias[criterio]))
##       estrategia = resumo_estrategias[linha_estrategia, "Lever"]
##       estrategia
##     }, analisar_ensemble_com_melhor_estrategia = function(ensemble, dados_regret, var_cenarios, var_
##
##
##       ensemble = as.data.frame(ensemble)
##       dados_regret = as.data.frame(dados_regret)
##
##
##       dados_regret["MelhorEstrategia"] = dados_regret[var_resposta] == dados_regret$MaximoPorScenari
##
##       linhas_melhores_estrategias = which(dados_regret[var_resposta] == dados_regret$MaximoPorScenar
##
##       variaveis = c(var_cenarios, var_estrategias, var_resposta)
##
##       melhores_estrategias = as.data.frame(dados_regret[linhas_melhores_estrategias, variaveis])
##
##       ensemble_com_melhor_estrategia = dplyr::inner_join(ensemble, melhores_estrategias)
##
##       ensemble_com_melhor_estrategia["EstrategiaCandidata"] = ensemble_com_melhor_estrategia[var_est
##
##       #ensemble_com_melhor_estrategia = as.factor(ensemble_com_melhor_estrategia[var_estrategias])
##
##       ensemble_com_melhor_estrategia
##
##     }, obter_df_vulnerabilidade = function(results, estrategia_candidata, variavel_resposta = "sNPVP
##       if(sentido_vulnerabilidade == ">=") {
##         results$AnaliseRegret$Dados$CasoInteresse = as.numeric(results$AnaliseRegret$Dados[,variavel_
##       } else {
##         results$AnaliseRegret$Dados$CasoInteresse = as.numeric(results$AnaliseRegret$Dados[,variavel_
##       }
##
##       # Obter Ensemble com Dados Simulados:
##       ensemble_e_resultados = dplyr::inner_join(as.data.frame(results$Ensemble), results$AnaliseRegr
##
##       ensemble_e_resultados = ensemble_e_resultados[which(ensemble_e_resultados$Lever == estrategia_
##
##       # Retirar NAs do Ensemble
##       ensemble_e_resultados = na.omit(ensemble_e_resultados)
##
##       parametros_completos = readxl::read_xlsx(planilha_inputs, sheet = "params")
##
##       variaveis_incertas = parametros_completos$Variavel[which(parametros_completos$Tipo=="Incerto")]
```

```
##
##       x = ensemble_e_resultados[,c("Scenario", "Lever",variaveis_incertas)]
##       y = as.numeric(ensemble_e_resultados$CasoInteresse)
##
##       data.frame(CasoInteresse = y, x)
##     }, obter_df_diff_media_casos_interesse = function(df_vulnerabilidade) {
##       medias_interesse = df_vulnerabilidade %>% dplyr::filter(CasoInteresse == 1) %>% dplyr::select(-
##
##       medias_global = df_vulnerabilidade %>% dplyr::select(-CasoInteresse, -Scenario, -Lever)  %>% dp
##
##       max_global = df_vulnerabilidade %>% dplyr::select(-CasoInteresse, -Scenario, -Lever) %>% dplyr
##
##       min_global = df_vulnerabilidade %>% dplyr::select(-CasoInteresse, -Scenario, -Lever) %>% dplyr
##
##       range_global = max_global - min_global
##
##       medias_dif = (medias_interesse - medias_global) / range_global
##
##       v_medias_analisadas = colnames(medias_dif)
##
##       v_medias_dif = unname(t(medias_dif)[,1])
##
##       v_medias_global = unname(t(medias_global)[,1])
##
##       v_range_global = unname(t(range_global)[,1])
##
##       v_medias_interesse = unname(t(medias_interesse)[,1])
##
##       ordem = order(abs(v_medias_dif), decreasing = TRUE)
##
##       df_analise_medias = data.frame(
##         Ranking = 1:length(v_medias_analisadas),
##         Variavel = v_medias_analisadas[ordem],
##         DifMediaRelativa = v_medias_dif[ordem],
##         MediaCasosInteresse = v_medias_interesse[ordem],
##         MediaGlobal = v_medias_global[ordem],
##         Range = v_range_global[ordem]
##       )
##     }, plot_violino_casos_interesse_por_variavel  = function(df_vulnerabilidade, variavel, nome_amiga
##       call_grafico = substitute(
##         expr = ggplot2::ggplot(df_vulnerabilidade, aes(factor(CasoInteresse), Variavel)) + geom_violi
##         ,env = list(Variavel = as.name(variavel))
##       )
##
##       p <- eval(call_grafico) + xlab("Estratégia Vulnerável") + ylab(nome_amigavel_var)
##       p
##     }, plot_dispersao_casos_interesse_por_variavel  = function(df_vulnerabilidade, variavel1, nome_a
##       call_grafico = substitute(
##         expr =  ggplot(as.data.frame(df_vulnerabilidade), aes(x=Variavel1, y=Variavel2, color = as.fa
##         ,env = list(Variavel1 = as.name(variavel1), Variavel2 = as.name(variavel2))
##       )
##
##       p =  eval(call_grafico)
##
```

```
##       p = p + geom_point() + scale_color_manual(values = c("blue", "red"), name = "Caso de Interesse
##
##       p = p + xlab(nome_amigavel_var1) + ylab(nome_amigavel_var2)
##
##       p
##
##     }, library(dplyr), selecionar_ultimo_periodo = function(dados_simulacao, var_tempo) {
##       call = substitute(
##         expr = dados_simulacao %>% dplyr::filter(Tempo == max(Tempo)),
##         env = list(Tempo = as.name(var_tempo)))
##       eval(call)
##     }, selecionar_ultimo_periodo = function(dados_simulacao, var_tempo) {
##       call = substitute(
##         expr = dados_simulacao %>% dplyr::filter(Tempo == max(Tempo)),
##         env = list(Tempo = as.name(var_tempo)))
##       eval(call)
##     }, calcular_maximo_por_variavel = function(var_resposta, var_group, dados) {
##       call = substitute(
##         expr = {dplyr::group_by(dados, VarGroup) %>%
##             dplyr::summarise(Maximo = max(VarResposta))
##         }
##         ,
##         env = list(VarGroup = as.name(var_group), VarResposta = as.name(var_resposta)))
##
##       max_variavel_resposta = eval(call)
##
##       dados_join = dplyr::inner_join(dados, max_variavel_resposta)
##
##       dados_join$Maximo
##     }, calcular_minimo_por_variavel = function(var_resposta, var_group, dados) {
##       call = substitute(
##         expr = {dplyr::group_by(dados, VarGroup) %>%
##             dplyr::summarise(Minimo = min(VarResposta))
##         }
##         ,
##         env = list(VarGroup = as.name(var_group), VarResposta = as.name(var_resposta)))
##
##       max_variavel_resposta = eval(call)
##
##       dados_join = dplyr::inner_join(dados, max_variavel_resposta)
##
##       dados_join$Minimo
##     }, completeFun <- function(data, desiredCols) {
##       completeVec <- complete.cases(data[, desiredCols])
##       return(data[completeVec, ])
##     }, getCost<-function(p, modelo, dados_calibracao){
##
##       #browser()
##       output_modelo = solve_modelo_dissertacao(parametros = p, modelo = modelo, simtime = SIM_TIME)
##       #output_modelo <- solve_modelo_dissertacao(parametros, modelo, simtime = SIM_TIME)
##       #http://www.inside-r.org/packages/cran/FME/docs/modCost
##       #browser()
##
##       cost <- modCost(obs=dados_calibracao, model=output_modelo)
```

```
##
##       #browser()
##
##       return(cost)
##
##     }, adicionar_erro_ao_ensemble = function(results, variavel_calibracao, planilha_calibracao, leve
##
##       dados_calibracao <- as.data.frame(read_xlsx(path = planilha_calibracao, sheet = "Plan1"))
##
##       variaveis_a_utilizar_modelo = c(opcoes$VarCenarios, opcoes$VarTempo, variavel_calibracao)
##
##       variaveis_a_utilizar_dados = c(opcoes$VarTempo, variavel_calibracao)
##
##       # Se um lever não foi informado, usar o primeiro do ensemble
##       if(missing(lever)){
##         # Usar o primeiro lever que eu achar
##         lever = results$DadosSimulados$Lever[1]
##       }
##
##       dados_modelo = results$DadosSimulados[which(results$DadosSimulados$Lever == lever),variaveis_a
##
##       cenarios = unique(dados_modelo$Scenario)
##
##       # Função para Obter Estatísticas de Fit
##       obter_estatisticas_fit = function(cenario, dados_modelo = dados_modelo , dados_calibracao = da
##         modcost = FME::modCost(model = dados_modelo[which(dados_modelo$Scenario == cenario),],
##                                obs = dados_calibracao[,variaveis_a_utilizar_dados])
##
##         # Soma dos Erros Quadrados
##         SumOfSquareResiduals = modcost$model
##
##         # Mean Square error
##         MeanSquareError = SumOfSquareResiduals / modcost$var$N
##
##         # Mean Absolute Error
##         MeanAbsoluteError = sum(abs(modcost$residuals$res)) / modcost$var$N
##
##         # Mean Absolute Percent Error
##         MeanAbsolutePercentError = (sum(abs(modcost$residuals$res) / abs(modcost$residuals$obs))) / n
##
##         # Thiel Statistics: Morecroft (2007), pg. 399.
##         UM_ThielBiasDiffMeans =  ((mean(modcost$residuals$mod, na.rm = TRUE) - mean(modcost$residuals
##
##         US_ThielUnequalVariation = ((sd(modcost$residuals$mod, na.rm = TRUE) - sd(modcost$residuals$o
##
##         UC_ThielUnequalCovariation = (1 / MeanSquareError) * (sd(modcost$residuals$mod, na.rm = TRUE)
##
##         stats_fit = c(SumOfSquareResiduals = SumOfSquareResiduals,
##                       MeanSquareError = MeanSquareError,
##                       MeanAbsoluteError = MeanAbsoluteError,
##                       MeanAbsolutePercentError = MeanAbsolutePercentError,
##                       UM_ThielBiasDiffMeans = UM_ThielBiasDiffMeans,
##                       US_ThielUnequalVariation = US_ThielUnequalVariation,
##                       UC_ThielUnequalCovariation = UC_ThielUnequalCovariation)
```

```
##         stats_fit
##       }
##
##       stats_fit = obter_estatisticas_fit(cenario = 1, dados_modelo = dados_modelo, dados_calibracao =
##
##       stats_fit = lapply(1:length(cenarios), obter_estatisticas_fit, dados_modelo = dados_modelo, dad
##       stats_fit = do.call(rbind, stats_fit)
##
##       ensemble = cbind(results$Ensemble, stats_fit)
##
##       ensemble
##
##       # Só a partir deste ponto os calculos são realizados.
##       # SomaSSR =  do.call(rbind, lapply(results$Ensemble[,opcoes$VarCenarios], obter_custo_cenario))
##       # colnames(SomaSSR) = c("SomaSSR")
##       #
##       # # Retornando o Ensemble com o Erro
##       # cbind(results$Ensemble, SomaSSR)
##
##     }, gerar_grafico_superficie = function(dados_ultimo_ano,variaveis, estrategia) {
##       dadosplot = subset.data.frame(dados_ultimo_ano, (Lever == estrategia))
##
##       dadosplot = dadosplot[variaveis]
##
##       dadosplot = as.matrix(dadosplot)
##
##       names = colnames(dadosplot)
##
##       s = interp(dadosplot[,1],dadosplot[,2],dadosplot[,3])
##
##       names(s) = names
##
##       # Plotando a População Final
##       f <- list(
##         family = "Courier New, monospace",
##         size = 18,
##         color = "#7f7f7f"
##       )
##       x <- list(
##         title = "Taxa Nascimento",
##         titlefont = f
##       )
##       y <- list(
##         title = "TaxaMorte",
##         titlefont = f
##       )
##       z <- list(
##         title = "Populacao",
##         titlefont = f
##       )
##
##       plot_ly(x = s[[1]], y = s[[2]], z = s[[3]]) %>% add_surface() %>% layout(xaxis = x, yaxis = y)
##
##     }, plot_clientes_uma_estrategia = function(dados, estrategia) {
```

```
##       gr2_dados = subset(dados, (Lever == estrategia))
##       ggplot2::ggplot(gr2_dados,
##                       aes(x=Tempo, y=Adopters, color=factor(Lever), group=Scenario)) +
##         geom_line() +
##         ylab("Clientes") +
##         xlab("Tempo") +
##         labs(color = "Estratégia")
##     }, plot_cash_uma_estrategia = function(dados, estrategia) {
##       gr2_dados = subset(dados, (Lever == estrategia))
##       ggplot2::ggplot(gr2_dados,
##                       aes(x=Tempo, y=Cash, color=factor(Lever), group=Scenario)) +
##         geom_line() +
##         ylab("Valor Presente") +
##         xlab("Tempo") +
##         labs(color = "Estratégia")
##     }, plot_linha_uma_variavel_ensemble_uma_estrategia = function(dados, variavel, nome_amigavel_vari
##
##       gr2_dados = subset(dados, (Lever %in% estrategia))
##
##       call_grafico = substitute(
##         expr = ggplot2::ggplot(gr2_dados, aes(x= time, y= Variavel, color=factor(Lever) , group= into
##                                               )),
##         env = list(Variavel = as.name(variavel))
##       )
##
##       p <- eval(call_grafico)
##
##       p +
##         geom_line() +
##         ylab(nome_amigavel_variavel) +
##         xlab("Tempo (anos)") +
##         theme(legend.position="bottom")  +
##         labs(color = "Estratégia")
##     }, plot_linha_uma_variavel_ensemble = function(dados, variavel, nome_amigavel_variavel, estrategi
##
##       levers_no_ensemble = unique(dados$Lever)
##
##       # Caso exista mais de uma estratégia, usar somente a primeira.
##       if(length(levers_no_ensemble) > 1) {
##         dados = subset(dados, Lever == 1)
##       }
##
##       call_grafico = substitute(
##         expr = ggplot2::ggplot(dados, aes(x= time, y= Variavel, color=Scenario, group= Scenario
##         )),
##         env = list(Variavel = as.name(variavel))
##       )
##
##       p <- eval(call_grafico)
##
##       p +
##         geom_line() +
##         ylab(nome_amigavel_variavel) +
##         xlab("Tempo (anos)") +
```

```
##         theme(legend.position="bottom")  +
##         labs(color = "Caso")
##     }, plot_linha_uma_variavel = function(dados, variavel, nome_amigavel_variavel) {
##
##       call_grafico = substitute(
##         expr = ggplot2::ggplot(dados, aes(x= time, y= Variavel)),
##         env = list(Variavel = as.name(variavel))
##       )
##
##       p <- eval(call_grafico)
##
##       p +
##         geom_line() +
##         ylab(nome_amigavel_variavel) +
##         xlab("Tempo (anos)") +
##         theme(legend.position="bottom")
##     }, plot_linha_duas_variaveis = function(dados, variavel1, nome_amigavel_variavel1, variavel2, nor
##
##       p <- ggplot2::ggplot(dados, aes(x = time))
##
##       call_variavel1 = substitute(
##         expr = p + geom_line(aes(y = Variavel, colour = NomeVariavel)),
##         env = list(Variavel = as.name(variavel1), NomeVariavel = nome_amigavel_variavel1)
##       )
##
##       razaovariavel = (max(dados[,variavel1]) - min(dados[,variavel1])) / (max(dados[,variavel2]) - m
##
##       p <- eval(call_variavel1)
##
##       call_variavel2 = substitute(
##         expr = p + geom_line(aes(y = Variavel * Razao, colour = NomeVariavel)),
##         env = list(Variavel = as.name(variavel2), NomeVariavel = nome_amigavel_variavel2, Razao = ra
##       )
##
##       p <- eval(call_variavel2)
##
##       # now adding the secondary axis, following the example in the help file ?scale_y_continuous
##       # and, very important, reverting the above transformation
##       p <- p + scale_y_continuous(sec.axis = sec_axis(~./razaovariavel, name = nome_amigavel_variavel
##
##       # modifying colours and theme options
##       p <- p + scale_colour_manual(values = c("blue", "red"))
##       p <- p + labs(y = nome_amigavel_variavel1,
##                     x = "Tempo (anos)",
##                     colour = "Variáveis")
##
##       p <- p + theme(legend.position="bottom")
##
##       p
##
##     }, plot_taxa_adocao_uma_estrategia = function(dados, estrategia) {
##       gr2_dados = subset(dados, (Lever == estrategia))
##       ggplot2::ggplot(gr2_dados,
##                       aes(x=Tempo, y=Adoption_Rate, color=factor(Lever), group=Scenario)) +
```

```
##         geom_line() +
##         ylab("Taxa de Adoção") +
##         xlab("Tempo (anos)") +
##         labs(color = "Estratégia")
##     }, grafico_whisker_por_lever = function(dados_regret, variavel) {
##       dados_por_estrategia = dplyr::group_by(dados_regret, Lever)
##
##       dados_por_estrategia$Lever = as.factor(dados_por_estrategia$Lever)
##
##       # Gerando Grafico da Variável de Perda de Oportunidade
##       call_grafico = substitute(
##         expr = ggplot(dados_por_estrategia, aes(y = Variavel,x = Lever, group = Lever)),
##         env = list(Variavel = as.name(variavel))
##       )
##
##       p <- eval(call_grafico)
##       p + geom_boxplot()
##     }, plot_fronteira_tradeoff_estrategia = function(results, opcoes = opcoes) {
##
##       dados_cenario = results$DadosUltimoPeriodo %>% dplyr::filter(AdvertisingCost  < 5.727e+04 & Ave
##
##       analise_regret_cenario = calcular_e_resumir_regret(dados = dados_cenario, var_resposta = opcoes
##
##       variavel_comparacao = paste(opcoes$VarResposta,opcoes$VarCriterio, sep = "")
##
##       variaveis_grafico_regret = c(opcoes$VarEstrategias, variavel_comparacao)
##
##       regret_todos_os_futuros = results$AnaliseRegret$ResumoEstrategias[variaveis_grafico_regret]
##
##       regret_todos_os_futuros = as.data.frame(regret_todos_os_futuros)
##
##       names(regret_todos_os_futuros) = c(opcoes$VarEstrategias, "PerdaOportunidadeTodosOsCenarios")
##
##       # names(regret_todos_os_futuros[variaveis_grafico_regret]) = c(opcoes$VarEstrategias, paste(var
##
##       regret_cenario = analise_regret_cenario$ResumoEstrategias[variaveis_grafico_regret]
##
##       regret_cenario = as.data.frame(regret_cenario)
##
##       names(regret_cenario) = c(opcoes$VarEstrategias, "PerdaOportunidadeNoCenario")
##
##       dados_join = dplyr::left_join(regret_todos_os_futuros, regret_cenario)
##
##       plot_ly(data = dados_join, x = ~PerdaOportunidadeTodosOsCenarios, y = ~PerdaOportunidadeNoCenar
##
##     }, plot_estrategias_versus_incertezas = function(ensemble_analisado, incertezas, binario = TRUE)
##       ensemble_analisado$EstrategiaCandidata = as.factor(ensemble_analisado$EstrategiaCandidata)
##
##       ensemble_analisado$Lever = as.factor(ensemble_analisado$Lever)
##
##       p = if(binario) {
##         GGally::ggpairs(ensemble_analisado, columns = incertezas, aes(colour = EstrategiaCandidata, a
##       } else {
##         GGally::ggpairs(ensemble_analisado, columns = incertezas, aes(colour = Lever, alpha = 0.7))
```

```
##       }
##
##       p
##     }, plot_landscape_futuros_plausiveis = function(results, estrategia, variavelresp, nomeamigavel_v
##
##       ensemble_e_resultados = dplyr::inner_join(as.data.frame(results$Ensemble), results$AnaliseRegr
##
##       ensemble_e_resultados = ensemble_e_resultados[which(ensemble_e_resultados$Lever == estrategia)
##
##       var_x = ensemble_e_resultados[,variavel1]
##       var_y = ensemble_e_resultados[,variavel2]
##       var_z = ensemble_e_resultados[,variavelresp]
##
##       my.df.interp <- interp(x = var_x, y = var_y, z = var_z, nx = 30, ny = 30)
##       my.df.interp.xyz <- as.data.frame(interp2xyz(my.df.interp))
##       names(my.df.interp.xyz) <- c(variavel1, variavel2, variavelresp)
##
##       my.df.interp.xyz = my.df.interp.xyz[complete.cases(my.df.interp.xyz),]
##
##
##       call_plot = substitute(
##         expr = ggplot(data = my.df.interp.xyz, aes(x = Variavelx, y = Variavely)) + geom_tile(aes(fi
##         env = list(Variavelx = as.name(variavel1), Variavely = as.name(variavel2), Variavelz = as.na
##       )
##
##       p <- eval(call_plot)
##       p <- p + xlab(n_variavel1) +
##         ylab(n_variavel2) +
##         labs(fill = nomeamigavel_variavelresp) +
##         ggtitle(label = paste("Estratégia", estrategia)) +
##         theme(plot.title = element_text(hjust = 0.5))
##       p
##     }, plot_grid_estrategias_casos_vpl = function(results) {
##       plot<-ggplot(results$DadosUltimoPeriodo, aes(Scenario, Lever, fill = sNPVProfit1)) +
##         geom_tile(colour="gray20", size=1.5, stat="identity") +
##         scale_fill_viridis(option="D") +
##         scale_y_continuous(breaks=1:6)+
##         xlab("Caso Simulado") +
##         ylab("Estratégias") +
##         theme(
##           #plot.title = element_text(color="white",hjust=0,vjust=1, size=rel(2)),
##           plot.background = element_blank(),
##           panel.background = element_blank(),
##           #panel.border = element_rect(fill=NA,color="gray20", size=0.5, linetype="solid"),
##           panel.grid.major = element_blank(),
##           panel.grid.minor = element_blank(),
##           axis.line = element_blank(),
##           axis.ticks = element_blank(),
##           #axis.text = element_text(color="white", size=rel(1.5)),
##           axis.text.y  = element_text(hjust=1),
##           #legend.text = element_text(color="white", size=rel(1.3)),
##           #legend.background = element_rect(fill="gray20"),
##           legend.position = "right"
##           #legend.title=element_blank()
```

```
##         )
##
##       plot$labels$fill = "VPL"
##       plot
##     }, sdrdm.pairs_plot = function(data, lever, variables) {
##       dados_grafico = subset(data, Lever == lever)
##       dados_grafico = dados_grafico[variables]
##
##       pairs(dados_grafico)
##     }, gerar_grafico_curva_experiencia = function() {
##       aLCStrength = c(0.5, 0.75, 0.85, 0.95, 1)
##
##       aLCExponent = log(aLCStrength)/log(2)
##
##       aInitialProductionExperience = 1000
##
##       sCumulativeProduction = 1000:5000
##
##       learning_df = data.frame(Learning0.5 = (sCumulativeProduction/aInitialProductionExperience)^aL
##                                Learning0.75 = (sCumulativeProduction/aInitialProductionExperience)^al
##                                Learning0.85 = (sCumulativeProduction/aInitialProductionExperience)^al
##                                Learning0.95 = (sCumulativeProduction/aInitialProductionExperience)^al
##                                Learning1 = (sCumulativeProduction/aInitialProductionExperience)^aLCE
##
##       aInitialUnitFixedCost = 2000
##
##       aInitialUnitVariableCost = 1000
##
##       # aUnitFixedCost = aLearning * aInitialUnitFixedCost
##
##       aUnitVariableCost = data.frame(learning_df * aInitialUnitVariableCost, Producao = sCumulativeP
##
##       ggplot2::ggplot(aUnitVariableCost, aes(Producao)) +
##         geom_line(aes(y = Learning1, colour = "1")) +
##         geom_line(aes(y = Learning0.95, colour = "0.95")) +
##         geom_line(aes(y = Learning0.85, colour = "0.85")) +
##         geom_line(aes(y = Learning0.75, colour = "0.75")) +
##         geom_line(aes(y = Learning0.5, colour = "0.5")) +
##         ylab("Custo Variável de Produção") +
##         xlab("Produção Acumulada") +
##         labs(color = expression(Gamma))
##     }, obter_dados_fundamentos_us_fundamentals = function(api_us_fundamentals = "AzfxuwuOWMDrCA28nAE
##       codigos_api = paste(codigos, collapse = ",")
##
##       indicadores_api = paste(indicadores, collapse = ",")
##
##       call_api = paste(
##         "https://api.usfundamentals.com/v1/indicators/xbrl?",
##         "&companies=",
##         codigos_api,
##         "&period_type=yq",
##         "&token=",
##         api_us_fundamentals
##         , sep = ""
```

```
##       )
##
##       anos = 2011:2016
##
##       csv_dados_us_fundamentals <- RCurl::getURL(call_api)
##
##       dados_us_fundamentals = read.csv(textConnection(csv_dados_us_fundamentals))
##
##       # renomeando colunas
##       colnames(dados_us_fundamentals) = c("company_id", "indicator_id", anos)
##
##       # Gerando tabela de empresas
##       empresas_e_codigos = data.frame(company_id = codigos, empresa = empresas)
##
##       # Indicando nome da empresa
##       dados_us_fundamentals = dplyr::inner_join(dados_us_fundamentals, empresas_e_codigos)
##
##
##       # Escrevendo CSV para guardar os dados antes de filtrar:
##       write.csv2(x = dados_us_fundamentals, file = "./fundamentals-data/dados_us_fundamentals.csv")
##
##       # Filtrando só o indicador desejado, ou definindo um conjunto de indicadores
##       if (length(indicadores)> 0){
##         dados_us_fundamentals = subset(dados_us_fundamentals, indicator_id %in% indicadores)
##       } else {indicadores = unique(dados_us_fundamentals$indicator_id)}
##
##       dados_finais = dados_us_fundamentals %>% tidyr::gather(Ano, Valor, 3:8) %>% tidyr::spread(indi
##
##       dados_finais
##     }, obter_fundamentos_financeiros_quandl = function(company_code = "DDD") {
##
##       # Base de Dados: -
##       # https://www.quandl.com/data/SF0-Free-US-Fundamentals-Data
##       # Com algumas alterações é possível usar outras bases do quandl.
##
##       # Definindo Chave de Acesso - Esta chave pertence a Pedro Lima, não utilizar:
##       Quandl.api_key("RsCuvs4_WjRPP_zzSzfv")
##
##       prefixo_base = "SF0/"
##
##       variable_codes = c("REVENUE_MRY",
##                          "INVENTORY_MRY",
##                          "ASSETS_MRY",
##                          "CAPEX_MRY",
##                          "NETINC_MRY",
##                          "GP_MRY",
##                          "COR_MRY",
##                          "TANGIBLES_MRY",
##                          "EBT_MRY",
##                          "FCF_MRY",
##                          "INTANGIBLES_MRY",
##                          "NCFI_MRY",
##                          "NCFF_MRY",
##                          'NCFO_MRY',
```

```
##                         "RND_MRY",
##                         "EBITDA_MRY")
##
##
##       variable_names = c("Revenue",
##                         "Inventory",
##                         "Assets",
##                         "Capex",
##                         "NetIncome",
##                         "GrossProfit",
##                         "CostOfRevenue",
##                         "TangibleAssets",
##                         "EBT",
##                         "FreeCashFlow",
##                         "IntangibleAssets",
##                         "NetCashFlowFromInvestment",
##                         "NetCashFlowFromFinancing",
##                         'NetCashFlowFromOperations',
##                         "ResearchAndDevelopmentExpenses",
##                         "EBITDA")
##
##
##       variable_descriptions = c(
##         "[Revenues]: Amount of Revenue recognized from goods sold, services rendered, insurance premi
##         ,"[Inventory]: A component of [ASSETS] representing the amount after valuation and reserves
##         ,"[Total Assets]: Sum of the carrying amounts as of the balance sheet date of all assets tha
##         ,"[Capital Expenditure]: A component of [NCFI] representing the net cash inflow (outflow) as
##         ,"[Net Income]: The portion of profit or loss for the period, net of income taxes, which is
##         ,"[Gross Profit]: Aggregate revenue [REVENUE] less cost of revenue [COR] directly attributabl
##         ,"[Cost of Revenue]: The aggregate cost of goods produced and sold and services rendered duri
##         ,"[Tangible Asset Value]: The value of tangibles assets calculated as the difference between
##         ,"[Earnings before Tax]: Earnings Before Tax is calculated by adding [TAXEXP] back to [NETINC
##         ,"[Free Cash Flow]: Free Cash Flow is a measure of financial performance calculated as [NCFO]
##         ,"[Goodwill and Intangible Assets]: A component of [ASSETS] representing the carrying amounts
##         ,"[Net Cash Flow from Investing]: A component of [NCF] representing the amount of cash inflow
##         ,"[Net Cash Flow from Financing]: A component of [NCF] representing the amount of cash inflow
##         ,"[Net Cash Flow from Operations]: A component of [NCF] representing the amount of cash inflo
##         ,"[Research and Development Expense]: A component of [OPEX] representing the aggregate costs
##         ,"[Earnings Before Interest, Taxes & Depreciation Amortization (EBITDA)]: EBITDA is a non-GA
##       )
##
##
##       df_variaveis = data.frame(
##         VariableCodes = variable_codes,
##         VariableNames = variable_names,
##         VariableDescriptions = variable_descriptions
##       )
##
##       sep = "_"
##
##       queries = paste(prefixo_base, company_code,sep, variable_codes, sep = "")
##
##       list_company = list()
##       for (q in queries){
```

```
##         message(paste("Queriying Quandl for variable", q))
##         qnumber = which(queries == q)
##         list_company[[variable_names[qnumber]]] = Quandl(q, collapse="annual", start_date="1900-01-0
##       }
##
##       df_company = data.frame(time = as.vector(time(list_company[[1]])),
##                               as.data.frame(list_company))
##
##       ## Salvar Dados Coletados
##       write.csv2(df_company, file = paste("./fundamentals-data/financial_data", company_code, ".csv"
##
##       message(paste("Finalizada coleta de dados da empresa", company_code))
##
##       ## Gerar List com descrição das variáveis e resultados
##       list(Variaveis = df_variaveis, Dados = df_company)
##     }, mordm.mark.box <- function(box, mean, mass) {
##       result <- mordm.mark.rule(function(x) {
##         names <- colnames(box)
##         all(sapply(1:length(names), function(i) x[names[i]] >= box[1,names[i]] & x[names[i]] <= box[1
##       })
##
##       attr(result, "mean") <- mean
##       attr(result, "mass") <- mass
##       attr(result, "box") <- box
##       return(result)
##     }, mordm.mark.rule <- function(condition) {
##       mark <- condition
##       class(mark) <- "mark"
##       return(mark)
##     }, mordm.mark.points <- function(points) {
##       mordm.mark.rule(function(x) {
##         # This is a fast implementation to determine if points contains x
##         for (i in 1:nrow(points)) {
##           match <- TRUE
##
##           for (j in 1:ncol(points)) {
##             if (points[i,j] != x[j]) {
##               match <- FALSE
##               break
##             }
##           }
##
##           if (match) {
##             return(TRUE)
##           }
##         }
##
##         return(FALSE)
##
##         # This is the cleaner but slow version
##         #any(apply(points, 1, function(y) isTRUE(all.equal(x, y))))
##       })
##     }, mordm.mark.selection <- function() {
##       cat("Use the mouse to select the points in the plot\n")
```

```
##       flush.console()
##
##       set <- get("current.set", mordm.globals)
##       selection <- selectpoints3d(value=FALSE)
##
##       cat("Selected ")
##       cat(nrow(selection))
##       cat(" points!\n")
##
##       return(mordm.mark.points(set[selection[,"index"],]))
##     }, mordm.mark.box <- function(box, mean, mass) {
##       result <- mordm.mark.rule(function(x) {
##         names <- colnames(box)
##         all(sapply(1:length(names), function(i) x[names[i]] >= box[1,names[i]] & x[names[i]] <= box[1
##       })
##
##       attr(result, "mean") <- mean
##       attr(result, "mass") <- mass
##       attr(result, "box") <- box
##       return(result)
##     }, mordm.mark.union <- function(...) {
##       mordm.mark.rule(function(x) {
##         # faster version with shortcircuiting
##         rules <- unlist(list(...))
##
##         for (rule in rules) {
##           if (rule(x)) {
##             return(TRUE)
##           }
##         }
##
##         return(FALSE)
##
##         # cleaner version without shortcircuiting
##         #any(sapply(unlist(list(...)), function(rule) rule(x)))
##       })
##     }, mordm.select.indices <- function(set, marking, not=FALSE, or=FALSE) {
##       if (is.list(marking)) {
##         indices <- rep(TRUE, nrow(set))
##
##         for (mark in marking) {
##           if (or) {
##             indices <- indices | mordm.select.indices(set, mark, not, or)
##           } else {
##             indices <- indices & mordm.select.indices(set, mark, not, or)
##           }
##         }
##       } else if (is.function(marking)) {
##         indices <- apply(set, 1, marking)
##       } else {
##         stop("Markings must be a function or a list of functions")
##       }
##
##       if (not) {
```

```
##         indices <- !indices
##       }
##
##       return(indices)
##     }, analyze.prim <- function(factors, response, bounds=NULL, which.box=1, show.plot=TRUE, method=
##       if (method == "prim") {
##         box <- prim.box(factors, response, ...)
##
##         marks <- lapply(1:box$num.hdr.class, function(i) {
##           i <- eval(i)
##           colnames(box$box[[i]]) <- colnames(factors)
##           mordm.mark.box(box$box[[i]], box$y.fun[i], box$mass[i])
##         })
##
##         if (is.null(bounds)) {
##           bounds <- apply(factors, 2, range)
##         }
##
##         dummy.data <- list()
##         attr(dummy.data, "nvars") <- ncol(factors)
##         attr(dummy.data, "bounds") <- bounds
##         mordm.plot.box(dummy.data, marks[[which.box]])
##
##         # compute density and coverage of the box
##         varargs <- list(...)
##
##         if (is.null(varargs$threshold.type) || varargs$threshold.type==0) {
##
##         } else if (varargs$threshold.type == -1) {
##           threshold <- mean(response)
##           total.interesting = sum(response <= threshold)
##
##           captured.indices <- mordm.select.indices(factors, mordm.mark.union(marks))
##           captured.interesting = sum(response[captured.indices] <= threshold)
##
##           cat("Coverage: ")
##           cat(captured.interesting / total.interesting)
##           cat("\n")
##           cat("Density: ")
##           cat(captured.interesting / length(captured.indices))
##           cat("\n")
##         } else {
##           threshold <- mean(response)
##           total.interesting = sum(response >= threshold)
##
##           captured.indices <- mordm.select.indices(factors, mordm.mark.union(marks))
##           captured.interesting = sum(response[captured.indices] >= threshold)
##
##           cat("Coverage: ")
##           cat(captured.interesting / total.interesting)
##           cat("\n")
##           cat("Density: ")
##           cat(captured.interesting / length(captured.indices))
##           cat("\n")
```

```
##         }
##
##         invisible(marks)
##       } else if (method == "sdprim") {
##         if (packageVersion("sdtoolkit") > '2.31') {
##           warning("Newer version of sdtoolkit have known errors.  Please use version 2.31 or before.
##         } else if (Sys.getenv("RSTUDIO") == "1") {
##           warning("sdtoolkit often causes RStudio to crash.  Please run PRIM from the console.")
##         }
##
##         require(sdtoolkit)
##         sdprim(factors, response, ...)
##       } else {
##         warning("Unknown PRIM method, must be 'prim' or 'sdprim'")
##       }
##     }, analyze.cart <- function(factors, response) {
##       extended.set <- data.frame(factors, response=response)
##       param_names <- colnames(factors)
##       names(extended.set) <- c(param_names, "response")
##       formula <- sprintf("`response` ~ %s", paste(sprintf("`%s`", param_names), collapse="+"))
##       fit <- rpart(as.formula(formula), data=extended.set, method="class")
##       fit <- prune(fit, cp=fit$cptable[which.min(fit$cptable[,"xerror"]),"CP"])
##       rpart.plot::prp(fit, type=2, extra=1)
##     })
```

10.2 Algoritmo para a Replicação dos Resultados

```
## expression(source('funcoes.R', encoding = 'UTF-8'), plots_width = 6,
##     plots_heigh = 3, USAR_DADOS_SALVOS = FALSE, opcoes_iniciais = list(
##       VarResposta = "sNPVProfit1",
##       VarCenarios = "Scenario",
##       VarEstrategias = "Lever",
##       N = 100,
##       VarTempo = "time",
##       VarCriterio = "RegretPercPercentil75",
##       SentidoCriterio = "min",
##       Paralelo = TRUE,
##       ModoParalelo = "FORK",
##       SimularApenasCasoBase = TRUE,
##       FullFactorialDesign = TRUE
##     ), opcoes = opcoes_iniciais, planilha_inputs = "./calibracao/params_calibracao_com_estrategia.xls
##     START<-2007, FINISH <-2027, STEP<-0.125, SIM_TIME <- seq(START, FINISH, by=STEP),
##     VERIFICAR_STOCKS = FALSE, VERIFICAR_CHECKS = FALSE, CHECK_PRECISION = 0.001,
##     BROWSE_ON_DIFF = TRUE, VERIFICAR_GLOBAL = FALSE, source('funcoes.R', encoding = 'UTF-8'),
##     resultados_casos_plausiveis = simularRDM_e_escolher_estrategia(inputs = planilha_inputs,
##                                                                    sdmodel = sdmodel,
##                                                                    opcoes = opcoes),
##     variavel_calibracao = "fIndustryOrderRate", nome_amigavel_variavel_calibracao = "Demanda Imp. 3D
##     resultados_casos_plausiveis$Inputs$Parametros, head(resultados_casos_plausiveis$Ensemble, 10),
##     head(resultados_casos_plausiveis$DadosSimulados, 20), cenarios_a_exibir_grafico = sample(1:opcoes
##     plot_demanda_pre_calibracao = plot_linha_uma_variavel_ensemble(dados = subset(resultados_casos_pl
##                                       variavel = variavel_calibracao,
##                                       nome_amigavel_variavel = nome_amigavel_variavel_calibracao) + g
```

```
##     plot_demanda_pre_calibracao = plot_demanda_pre_calibracao + annotate("text", x = 2017.2, y = max
##     ensemble_com_erro = adicionar_erro_ao_ensemble(results = resultados_casos_plausiveis, variavel_ca
##     ensemble_com_erro = as.data.frame(ensemble_com_erro), variaveis_analise_fit = c("SumOfSquareResi
##     variaveis_exibir_ensemble = c("Scenario", variaveis_analise_fit),
##     cenarios_a_exibir_tabela = sample(1:opcoes$N,size = 5), t(ensemble_com_erro[1:5,variaveis_exibir
##     histograma_erro_percentual = ggplot(ensemble_com_erro, aes(x=MeanAbsolutePercentError)) +
##                                    geom_histogram(aes(y=..density..), colour="black", fill="white")+
##                                    geom_density(alpha=.2, fill="#FF6666") +
##                                    xlab("Erro Médio Percentual") +
##                                    ylab("Densidade"), histograma_erro_percentual,
##     histograma_erro_medio_quadrado = ggplot(ensemble_com_erro, aes(x=MeanSquareError)) +
##       geom_histogram(aes(y=..density..), colour="black", fill="white")+
##       geom_density(alpha=.2, fill="#FF6666") +
##       xlab("Erro Médio Quadrado") +
##       ylab("Densidade"), histograma_erro_medio_quadrado, cenario_menor_erro = ensemble_com_erro[which
##     parametros_cenario_menor_erro = t(ensemble_com_erro[which(ensemble_com_erro[,opcoes$VarCenarios]=
##     parametros_cenario_menor_erro, time_points<-seq(from=1, to=length(SIM_TIME),by=1/STEP),
##     time_plot = seq(from=START, to=FINISH), resultados_exibir = dplyr::filter(resultados_casos_plausi
##     plot_cenario_base_e_historico <-ggplot()+
##       geom_point(data=dados_calibracao,size=1.5,aes(time,fIndustryOrderRate,colour="Data"))+
##       geom_line(data=resultados_exibir,size=1,aes(x=time,y=fIndustryOrderRate,colour="Model"))+
##       ylab("Demanda Total")+
##       xlab("Anos")+
##       scale_y_continuous(labels = comma)+
##       theme(legend.position="bottom")+
##       scale_colour_manual(name="",
##                           values=c(Data="red",
##                                    Model="blue"),
##                           labels=c("Dados",
##                                    "Modelo")), plot_cenario_base_e_historico,
##     percentil_utilizado_como_criterio = c(PercentilCriterio = 0.5),
##     percentil_ssr = quantile(ensemble_com_erro[,"SumOfSquareResiduals"], probs = c(percentil_utiliza
##     cenarios_quartis = ensemble_com_erro[which(ensemble_com_erro[,"SumOfSquareResiduals"] <= percenti
##     cenarios_considerados_plausiveis = cenarios_quartis, dados_calibracao$Scenario = 1000,
##     plot_cenarios_plausiveis = plot_linha_uma_variavel_ensemble(dados = resultados_casos_plausiveis$D
##                                      ,variavel = variavel_calibracao, nome_amigavel_variavel = nome_a
##     plot_cenarios_plausiveis, ensemble_a_simular = resultados_casos_plausiveis$Ensemble[which(result
##     ggplot(as.data.frame(ensemble_a_simular), aes(x=aReferencePopulation, y=aWOMStrength)) + geom_po
##     opcoes$SimularApenasCasoBase = FALSE, START<-2007, FINISH <-2027,
##     STEP<-0.125, SIM_TIME <- seq(START, FINISH, by=STEP), VERIFICAR_STOCKS = FALSE,
##     VERIFICAR_CHECKS = FALSE, CHECK_PRECISION = 0.001, BROWSE_ON_DIFF = TRUE,
##     VERIFICAR_GLOBAL = FALSE, source('funcoes.R', encoding = 'UTF-8'),
##     results = simularRDM_e_escolher_estrategia(inputs = planilha_inputs,
##                                                sdmodel = sdmodel,
##                                                opcoes = opcoes,
##                                                ensemble = ensemble_a_simular),
##     plots_rodada1 = list(
##       plot_estrategia1 = plot_linha_uma_variavel_ensemble(dados = results$DadosSimulados, variavel =
##       plot_1_e_candidata = plot_linha_uma_variavel_ensemble(dados = results$DadosSimulados, variavel
##       plot_preco_estrategia10 = plot_linha_uma_variavel_ensemble(dados = results$DadosSimulados, var
##       plot_estrategia_candidata = plot_linha_uma_variavel_ensemble(dados = results$DadosSimulados, va
##       plot_whisker_lever_perc_regret = grafico_whisker_por_lever(results$AnaliseRegret$Dados, variav
##       plot_whisker_lever_regret = grafico_whisker_por_lever(results$AnaliseRegret$Dados, variavel =
##       plot_whisker_lever_profit = grafico_whisker_por_lever(results$AnaliseRegret$Dados, variavel =
```

```
##     ), plots_rodada1$plot_estrategia_candidata, plots_rodada1$plot_1_e_candidata,
##     plots_rodada1$plot_preco_estrategia10, plots_rodada1$plot_estrategia_candidata,
##     plots_rodada1$plot_whisker_lever_regret, plots_rodada1$plot_whisker_lever_profit,
##     plots_rodada1$plot_whisker_lever_perc_regret, results$AnaliseRegret$ResumoEstrategias,
##     plots_rodada1$plot_whisker_lever_perc_regret, regret_perc_estrategia_candidata = results$Analisel
##     hist(regret_perc_estrategia_candidata), summary(regret_perc_estrategia_candidata),
##     quartis = quantile(regret_perc_estrategia_candidata, c(0.25, 0.5, 0.75)),
##     df_vulnerabilidade = obter_df_vulnerabilidade(results = results,
##                                                   estrategia_candidata = results$EstrategiaCandidata
##                                                   variavel_resposta = "sNPVProfit1RegretPerc" ,
##                                                   threshold = 0.2,
##                                                   planilha_inputs = planilha_inputs,
##                                                   sentido_vulnerabilidade = ">="),
##     ranking_variaveis_por_media = obter_df_diff_media_casos_interesse(df_vulnerabilidade = df_vulnera
##     ranking_variaveis_por_media, plot_violino_casos_interesse_por_variavel(df_vulnerabilidade = df_vu
##     plot_dispersao_casos_interesse_por_variavel(df_vulnerabilidade = df_vulnerabilidade, variavel1 =
##     plot_dispersao_casos_interesse_por_variavel(df_vulnerabilidade = df_vulnerabilidade, variavel1 =
##     plot_dispersao_casos_interesse_por_variavel(df_vulnerabilidade = df_vulnerabilidade, variavel1 =
##     n_variaveis_shortlist = 10, variaveis_shortlist = as.vector(ranking_variaveis_por_media$Variavel
##     y = df_vulnerabilidade$CasoInteresse, x = df_vulnerabilidade[,variaveis_shortlist],
##     library(party), cf1 <- cforest(y ~ . , data= x, control=cforest_unbiased(mtry=2,ntree=50)),
##     varimp(cf1), varimp(cf1, conditional=TRUE), varimpAUC(cf1),
##     library(relaimpo), lmMod <- lm(y ~ . , data = x), relImportance <- calc.relimp(lmMod, type = "lmg
##     calc.relimp(lmMod, rela = TRUE), sort(relImportance$lmg, decreasing=TRUE),
##     library(breakDown), br = broken(lmMod, new_observation = ensemble_e_resultados[18,]),
##     plot(br), library(earth), marsModel <- earth(y ~ ., data= x),
##     ev <- evimp(marsModel), plot(ev), base.mod <- lm(y ~ 1 , data= x),
##     all.mod <- lm(y ~ . , data= x), stepMod <- step(base.mod, scope = list(lower = base.mod, upper =
##     shortlistedVars <- names(unlist(stepMod[[1]])), shortlistedVars <- shortlistedVars[!shortlistedVa
##     print(shortlistedVars), library(Boruta), boruta_output <- Boruta(y ~ ., data=na.omit(x), doTrace=
##     boruta_signif <- names(boruta_output$finalDecision[boruta_output$finalDecision %in% c("Confirmed
##     print(boruta_signif), plot(boruta_output, cex.axis=.7, las=2, xlab="", main="Variable Importance
##     variavel_1 = boruta_signif[1], variavel_2 = boruta_signif[2],
##     landscape_estrategia = plot_landscape_futuros_plausiveis(
##       results, estrategia = estrategia_candidata,
##       variavelresp = variavel_resposta,
##       nomeamigavel_variavelresp = "LOF Percentual",
##       variavel1  = variavel_1,
##       n_variavel1 = "Sensibilidade ao Preço",
##       variavel2 = variavel_2,
##       n_variavel2 = "Utilização da Capacidade"
##     ), library(rpart), cart_fit = rpart(y ~., data = x, method = "class"),
##     printcp(cart_fit), plotcp(cart_fit), summary(cart_fit), cart_fit$variable.importance,
##     summary(cart_fit), rpart.plot::prp(fit, digits = 3, clip.right.labs = FALSE, varlen = 0),
##     library(prim), results.prim = prim.box(x = x[,1:2], y = y, threshold.type = 1, peel.alpha = 0.05
##     teste_prim_mordm = analyse.prim(factors = x, response = y, bounds=NULL, which.box=1, show.plot=TR
##     summary(results.prim, print.box = TRUE), plot(results.prim, splom=FALSE),
##     results.prim, plot_dispersao_casos_interesse_por_variavel(df_vulnerabilidade = df_vulnerabilidade
##     plot(results.prim, col="transparent"), plot(results.prim),
##     points(x), library(subgroup.discovery), subgroup.discovery::prim.,
##     results.prim = prim.box(x = x, y = y, threshold.type = 1, peel.alpha = 0.25, paste.alpha = 0.15,
##     summary(results.prim, print.box = TRUE), write.csv(ensemble_e_resultados),
##     arquivo_ensemble_e_resultados = "ensemble_e_resultados.rda",
##     arquivo_salvar = ensemble_e_resultados[,c(variavel_resposta,variaveis_incertas)],
```

```
##     save(arquivo_salvar, file = arquivo_ensemble_e_resultados),
##     library(sdtoolkit), sdtoolkit::sdprim(x = x, y = y), sdtoolkit::sd.start(),
##     assign("mordm.globals", new.env()), factors = x, response = y,
##     box = prim.box(x = x, y = y, threshold.type = 1, peel.alpha = 0.25, paste.alpha = 0.15, threshol
##     bounds = NULL, which.box=1, box <- prim.box(x = factors, y = response, threshold.type = 1, peel.a
##     marks <- lapply(1:box$num.hdr.class, function(i) {
##       i <- eval(i)
##       colnames(box$box[[i]]) <- colnames(factors)
##       mordm.mark.box(box$box[[i]], box$y.fun[i], box$mass[i])
##     }), if (is.null(bounds)) {
##       bounds <- apply(factors, 2, range)
##     }, dummy.data <- list(), attr(dummy.data, "nvars") <- ncol(factors),
##     attr(dummy.data, "bounds") <- bounds, mordm.plot.box(dummy.data, marks[[which.box]]),
##     varargs$threshold.type = -1, varargs <- list(...), if (is.null(varargs$threshold.type) || varargs
##
##     } else if (varargs$threshold.type == -1) {
##       threshold <- mean(response)
##       total.interesting = sum(response <= threshold)
##
##       captured.indices <- mordm.select.indices(factors, mordm.mark.union(marks))
##       captured.interesting = sum(response[captured.indices] <= threshold)
##
##       cat("Coverage: ")
##       cat(captured.interesting / total.interesting)
##       cat("\n")
##       cat("Density: ")
##       cat(captured.interesting / length(captured.indices))
##       cat("\n")
##     } else {
##       threshold <- mean(response)
##       total.interesting = sum(response >= threshold)
##
##       captured.indices <- mordm.select.indices(factors, mordm.mark.union(marks))
##       captured.interesting = sum(response[captured.indices] >= threshold)
##
##       cat("Coverage: ")
##       cat(captured.interesting / total.interesting)
##       cat("\n")
##       cat("Density: ")
##       cat(captured.interesting / length(captured.indices))
##       cat("\n")
##     }, invisible(marks), START<-0, FINISH<-10, STEP<-0.0625,
##     SIM_TIME <- seq(START, FINISH, by=STEP), VERIFICAR_STOCKS = FALSE,
##     VERIFICAR_CHECKS = FALSE, CHECK_PRECISION = 0.001, BROWSE_ON_DIFF = TRUE,
##     VERIFICAR_GLOBAL = FALSE, source('modelo-calibracao.R', encoding = 'UTF-8'),
##     arquivo_excel_stocks = carregar_inputs(arquivo_de_inputs = "../modelo-ithink/dados_ithink_excel_s
##     arquivo_excel_checks = carregar_inputs(arquivo_de_inputs = "../modelo-ithink/dados_ithink_excel_c
##     arquivo_excel_global = carregar_inputs(arquivo_de_inputs = "../modelo-ithink/dados_ithink_tudo.xl
##     dados_ithink_stocks  = arquivo_excel_stocks$ResultadosIthink %>% dplyr::select(-Months),
##     dados_ithink_checks  = arquivo_excel_checks$ResultadosIthink %>% dplyr::select(-Months),
##     dados_ithink_global = arquivo_excel_global$ResultadosIthink %>% dplyr::select(-Months),
##     variaveis_ithink_stocks = names(dados_ithink_stocks), variaveis_ithink_checks = names(dados_ithin
##     variaveis_globais_a_verificar = carregar_inputs(arquivo_de_inputs = "../modelo-ithink/variaveis_g
##     variaveis_globais_a_verificar = as.vector(variaveis_globais_a_verificar$ResultadosIthink$variavei
```

```
##     arquivo_parametros = "./calibracao/params_calibracao.xlsx",
##     parametros_completos = readxl::read_xlsx(arquivo_parametros, sheet = "params"),
##     parametros_cenariobase = t(parametros_completos[,"CenarioBase"])[1,],
##     names(parametros_cenariobase) = as.matrix(parametros_completos[,1]),
##     resultados_cenariobase = solve_modelo_dissertacao(parametros = parametros_cenariobase, modelo = s
##     START<-2007, FINISH<-2037, STEP<-0.0625, SIM_TIME <- seq(START, FINISH, by=STEP),
##     VERIFICAR_STOCKS = FALSE, VERIFICAR_CHECKS = FALSE, CHECK_PRECISION = 0.01,
##     BROWSE_ON_DIFF = TRUE, VERIFICAR_GLOBAL = FALSE, resultados_cenarioscalibracao = simularRDM_e_es
##     resultados_cenarioscalibracao$Ensemble = adicionar_erro_ao_ensemble(results = resultados_cenarios
##     dados_calibracao <- as.data.frame(read_xlsx(path = "./calibracao/dados_calibracao.xlsx", sheet =
##     hist(resultados_cenarioscalibracao$Ensemble[,"SumOfSquareResiduals"]),
##     quartil1_erro = quantile(resultados_cenarioscalibracao$Ensemble[,"SumOfSquareResiduals"], probs =
##     cenarios_quartis = resultados_cenarioscalibracao$Ensemble[which(resultados_cenarioscalibracao$Ens
##     cenario_menor_erro = resultados_cenarioscalibracao$Ensemble[which(resultados_cenarioscalibracao$E
##     time_points<-seq(from=1, to=length(SIM_TIME),by=1/STEP),
##     time_plot = seq(from=START, to=FINISH), resultados_exibir = dplyr::filter(resultados_cenarioscali
##     p1<-ggplot()+
##       geom_point(data=dados_calibracao,size=1.5,aes(time,fIndustryOrderRate,colour="Data"))+
##       geom_line(data=resultados_exibir,size=1,aes(x=time,y=fIndustryOrderRate,colour="Model"))+
##       ylab("Demanda Anual - Impressoras 3D > 5k USD")+
##       xlab("Anos")+
##       scale_y_continuous(labels = comma)+
##       theme(legend.position="bottom")+
##       scale_colour_manual(name="",
##                           values=c(Data="red",
##                                    Model="blue"),
##                           labels=c("Dados",
##                                    "Modelo")), p1, resultados_exibir_outros_parametros = resultados_e
##     p2<-ggplot()+
##       geom_point(data=dados_calibracao,size=1.5,aes(time,fIndustryOrderRate,colour="Data"))+
##       geom_line(data=resultados_exibir_outros_parametros,size=0.5,aes(x=time,y=fIndustryOrderRate,col
##       ylab("Demanda da Indústria")+
##       xlab("Anos")+
##       scale_y_continuous(labels = comma)+
##       theme(legend.position="bottom")+
##       scale_colour_manual(name="",
##                           values=c(Data="red",
##                                    Model="blue"),
##                           labels=c("Dados",
##                                    "Modelo")), p2, plot_linha_uma_variavel_ensemble(dados = resultado
##     View(resultados_exibir_outros_parametros), START<-0, FINISH<-10,
##     STEP<-0.0625, SIM_TIME <- seq(START, FINISH, by=STEP), VERIFICAR_STOCKS = FALSE,
##     VERIFICAR_CHECKS = FALSE, CHECK_PRECISION = 0.00001, BROWSE_ON_DIFF = FALSE,
##     source('modelo-calibracao.R', encoding = 'UTF-8'), results = simularRDM_e_escolher_estrategia(inp
##     save(results, file = "./rodada1/resultados.Rdata"), dplyr::group_by(results$DadosSimulados, Scena
##     results$DadosUltimoPeriodo, plots_rodada1 = list(
##       plot_estrategia1 = plot_linha_uma_variavel_ensemble(dados = results$DadosSimulados, variavel =
##       plot_1_e_candidata = plot_linha_uma_variavel_ensemble(dados = results$DadosSimulados, variavel
##       plot_preco_estrategia10 = plot_linha_uma_variavel_ensemble(dados = results$DadosSimulados, vari
##       plot_estrategia_candidata = plot_linha_uma_variavel_ensemble(dados = results$DadosSimulados, va
##       plot_whisker_lever_perc_regret = grafico_whisker_por_lever(results$AnaliseRegret$Dados, variave
##       plot_whisker_lever_regret = grafico_whisker_por_lever(results$AnaliseRegret$Dados, variavel = "
##       plot_whisker_lever_profit = grafico_whisker_por_lever(results$AnaliseRegret$Dados, variavel = "
##     ), mapply(ggsave, file=paste0("./images/", names(plots_rodada1), ".png"), plot=plots_rodada1, wid
```

```
##     ensemble_analisado = analisar_ensemble_com_melhor_estrategia(ensemble = results$Ensemble,
##                                                                  dados_regret = results$AnaliseRegre
##                                                                  var_cenarios = opcoes$VarCenarios,
##                                                                  var_estrategias = opcoes$VarEstrate
##                                                                  var_resposta = opcoes$VarResposta,
##                                                                  estrategia_candidata = results$Estra
##     parametros_lidos = readxl::read_xlsx("./rodada1/params.xlsx", sheet = "params"),
##     incertezas = parametros_lidos[which(!(parametros_lidos[,"Min"]==parametros_lidos[,"Max"])),"Varia
##     incertezas = c("aLCStrength", "aNormalCapacityUtilization", "aSensOfAttractToPrice", "aRatioOfFi
##     plot_estrategias_incertezas = plot_estrategias_versus_incertezas(ensemble_analisado = ensemble_a
##     ggsave(filename = "./images/plot_estrategias_incertezas.png", plot = plot_estrategias_incertezas
##     estrategia = results$EstrategiaCandidata, ensemble_e_resultados = dplyr::inner_join(as.data.fram
##     ensemble_e_resultados = ensemble_e_resultados[which(ensemble_e_resultados$Lever == estrategia),]
##     variaveis_incertas = colnames(results$Ensemble), variaveis_incertas = variaveis_incertas[which(va
##     variavel_resposta = "sNPV1ProfitRegret", class(ensemble_e_resultados),
##     ensemble_e_resultados$sNPVProfit1Regret, library(party),
##     cf1 <- cforest(ensemble_e_resultados[,variavel_resposta] ~ . , data= ensemble_e_resultados[,varia
##     varimp(cf1), varimp(cf1, conditional=TRUE), varimpAUC(cf1),
##     library(relaimpo), lmMod <- lm(ensemble_e_resultados$sNPVProfit1Regret ~ . , data = ensemble_e_re
##     relImportance <- calc.relimp(lmMod, type = "lmg", rela = TRUE),
##     calc.relimp(lmMod, rela = TRUE), sort(relImportance$lmg, decreasing=TRUE),
##     library(breakDown), br = broken(lmMod, new_observation = ensemble_analisado[18,]),
##     plot(br), library(earth), marsModel <- earth(ensemble_e_resultados$sNPVProfit1Regret ~ ., data=en
##     ev <- evimp(marsModel), plot(ev), base.mod <- lm(ensemble_e_resultados$sNPVProfit1Regret ~ 1 , da
##     all.mod <- lm(ensemble_e_resultados$sNPVProfit1Regret ~ . , data= ensemble_e_resultados[,variave
##     stepMod <- step(base.mod, scope = list(lower = base.mod, upper = all.mod), direction = "both", t
##     shortlistedVars <- names(unlist(stepMod[[1]])), shortlistedVars <- shortlistedVars[!shortlistedVa
##     print(shortlistedVars), library(Boruta), boruta_output <- Boruta(ensemble_e_resultados$sNPVProfi
##     boruta_signif <- names(boruta_output$finalDecision[boruta_output$finalDecision %in% c("Confirmed
##     print(boruta_signif), plot(boruta_output, cex.axis=.7, las=2, xlab="", main="Variable Importance
##     landscape_estrategia1 = plot_landscape_futuros_plausiveis(
##       results, estrategia = 2,
##       variavelresp = "sNPVProfit1Regret",
##       nomeamigavel_variavelresp = "VPL",
##       variavel1  = "aRatioOfFixedToVarCost",
##       n_variavel1 = "Sensibilidade ao Preço",
##       variavel2 = "aNormalCapacityUtilization",
##       n_variavel2 = "Utilização da Capacidade"
##     ), rmarkdown::render(input = "Capitulo-4.Rmd"), resultado_unico <- data.frame(ode(y=stocks, time
##                         parms=auxs, method="euler")), dados_fundamentos = obter_dados_fundamentos_us_f
##     plot_lucro_bruto_us_fundamentals = ggplot(dados_fundamentos, aes(x=Ano, y=GrossProfit, group=emp
##       geom_line(aes(color=empresa))+
##       geom_point(aes(color=empresa)) +
##       ylab(label = "Lucro Bruto"), fundamentos_ddd = obter_fundamentos_financeiros_quandl("DDD"),
##     fundamentos_mtls = obter_fundamentos_financeiros_quandl("PRLB"),
##     plot_receita_investimento_3dsystems = plot_linha_duas_variaveis(fundamentos_ddd$Dados, variavel1
##     plot_cash_net_income_3dsystems = plot_linha_duas_variaveis(fundamentos_ddd$Dados, variavel1 = "N
##     resultado_transposto = t(resultado_unico), write.csv2(x = resultado_transposto, file = "resultad
##     results = simularRDM_e_escolher_estrategia(inputs = "params.xlsx", sdmodel = sdmodel, opcoes = o
##     incertezas = c("aAdvertisingEffectiveness", "aContactRate", "aAdoptionFraction", "aAdvertisingCo
##     ensemble_analisado = analisar_ensemble_com_melhor_estrategia(ensemble = results$Ensemble, dados_
##     plot_estrategias_versus_incertezas(ensemble_analisado, incertezas),
##     lever = results$EstrategiaCandidata, variaveis = c("CashRegretPerc","CashRegret","Cash", "Averag
##     sdrdm.pairs_plot(data = results$AnaliseRegret$Dados, lever = 1, variables = variaveis),
```

```
##     grafico_whisker_por_lever(dados_regret = results$AnaliseRegret$Dados, variavel = "Cash"),
##     grafico_whisker_por_lever(dados_regret = results$AnaliseRegret$Dados, variavel = "Adopters"),
##     grafico_whisker_por_lever(dados_regret = results$AnaliseRegret$Dados, variavel = "CashRegretPerc
##     grafico_whisker_por_lever(dados_regret = results$AnaliseRegret$Dados, variavel = "CashRegret"),
##     assign("mordm.globals", new.env()), START<-0, FINISH<-40,
##     STEP<-0.0625, SIM_TIME <- seq(START, FINISH, by=STEP), VERIFICAR_STOCKS = FALSE,
##     VERIFICAR_CHECKS = FALSE, CHECK_PRECISION = 0.00001, BROWSE_ON_DIFF = FALSE,
##     source('modelo-calibracao.R', encoding = 'UTF-8'), arquivo_parametros = "./analise-sterman/params
##     parametros_completos = readxl::read_xlsx(arquivo_parametros, sheet = "params"),
##     parametros_sterman = t(parametros_completos[,"Sterman"])[1,],
##     names(parametros_sterman) = as.matrix(parametros_completos[,1]),
##     resultados_sterman = solve_modelo_dissertacao(parametros = parametros_sterman, modelo = sdmodel$
##     sterman_plots = list(
##       grafico_npv_sterman = plot_linha_uma_variavel(dados = resultados_sterman, variavel = "sNPVProf
##
##       grafico_preco_sterman = plot_linha_uma_variavel(dados = resultados_sterman, variavel = "sPrice
##
##       grafico_demanda_sterman = plot_linha_uma_variavel(dados = resultados_sterman, variavel = "fIndu
##
##       grafico_vpl_preco = plot_linha_duas_variaveis(dados = resultados_sterman, variavel1 = "sNPVPro
##
##       grafico_vpl_demanda = plot_linha_duas_variaveis(dados = resultados_sterman, variavel1 = "sNPVP
##
##     ), save(sdmodel, parametros_sterman, resultados_sterman, sterman_plots, file = "./analise-sterma
##     mapply(ggsave, file=paste0("./images/", names(sterman_plots), ".png"), plot=sterman_plots, width
```