UNIVERSIDADE ESTADUAL PAULISTA “JÚLIO DE MESQUITA FILHO”
CAMPUS EXPERIMENTAL DE SÃO JOÃO DA BOA VISTA
ENGENHARIA ELETRÔNICA E DE TELECOMUNICAÇÕES

CAIO CESAR SOUZA DE LUCA

ANÁLISE COMPARATIVA DE MÉTODOS NUMÉRICOS PARA SIMULAÇÃO DE TRANSITÓRIOS ELETROMAGNÉTICOS

TRABALHO DE CONCLUSÃO DE CURSO

SÃO JOÃO DA BOA VISTA
2020

CAIO CESAR SOUZA DE LUCA

ANÁLISE COMPARATIVA DE MÉTODOS NUMÉRICOS PARA SIMULAÇÃO DE TRANSITÓRIOS ELETROMAGNÉTICOS

Trabalho de Conclusão de Curso apresentado como requisito obrigatório à obtenção do título de bacharel em engenharia eletrônica e de telecomunicações, da Universidade Estadual Paulista "Júlio de Mesquita Filho"

Orientador: Prof. Dr. Afonso José do Prado
Coorientador: Prof. Dr. Elmer Mateus Gennaro

SÃO JOÃO DA BOA VISTA
2020

Luca, Caio Cesar Souza de

Análise comparativa de métodos numéricos para simulação de transitórios eletromagnéticos / Caio Cesar Souza de Luca -- São João da Boa Vista, 2020.

55 p.: il. color.

Trabalho de Conclusão de Curso (Bacharelado - Engenharia Eletrônica e de Telecomunicações) – Universidade Estadual Paulista (Unesp), Câmpus Experimental de São João da Boa Vista, São João da Boa Vista.

Orientador: Prof. Dr. Afonso José do Prado

Coorientador: Prof. Dr. Elmer Mateus Gennaro

Bibliografia

1. Linhas elétricas 2. Otimização matemática 3. Telecomunicações 4. Transitórios (Eletricidade)

CDD 23. ed. – 621.382

Ficha catalográfica elaborada pela Biblioteca-BJB

Bibliotecário responsável: João Pedro Alves Cardoso – CRB-8/9717

Dedico este trabalho aos meus familiares.
Mãe, seu cuidado e apoio me deram
esperança para seguir.
Pai, sua segurança me deu certeza de
que não estou sozinho em todos os
passos da minha vida.

UNIVERSIDADE ESTADUAL PAULISTA
“JÚLIO DE MESQUITA FILHO”
CÂMPUS EXPERIMENTAL DE SÃO JOÃO DA BOA VISTA
GRADUAÇÃO EM ENGENHARIA ELETRÔNICA E DE TELECOMUNICAÇÕES

TRABALHO DE CONCLUSÃO DE CURSO

ANÁLISE COMPARATIVA DE MÉTODOS NUMÉRICOS PARA SIMULAÇÃO DE TRANSITÓRIOS ELETROMAGNÉTICOS

Aluno: Caio Cesar Souza de Luca
Orientador: Prof. Dr. Afonso José do Prado
Coorientador: Prof. Dr. Elmer Mateus Gennaro

Banca Examinadora:
- Afonso José do Prado (Orientador)
- Leonardo da Silva Lessa (Examinador)
- Wilian Miranda dos Santos (Examinador)

A ata da defesa com as respectivas assinaturas dos membros encontra-se no prontuário do aluno (Expediente nº 24/2019)

AGRADECIMENTOS

Gostaria de agradecer à UNESP pela chance de obter tanto conhecimento.

Aos meus pais por me apoiarem em todos os momentos da minha vida.

Aos amigos pelos conselhos e auxílio em vários momentos da vida.

Ao meu orientador que sempre me instruiu com seu conhecimento e me guiou para o melhor caminho.

Aos meus professores pela dedicação e atenção.

À FAPESP pelo auxílio financeiro para o desenvolvimento da pesquisa.

“A tarefa não é tanto ver aquilo que ninguém viu, mas pensar o que ninguém ainda pensou sobre aquilo que todo mundo vê.”

(Arthur Schopenhauer)

RESUMO

Este trabalho tem como base o desenvolvimento e análise de otimizações de dois modelos numéricos que representam linhas de transmissão. A otimização tem como ideia diminuir a memória de processamento utilizada nesses modelos e também reduzir o tempo de processamento delas. As análises apresentadas neste trabalho não estão relacionadas à minimização das oscilações numéricas (oscilações de Gibbs). Em relação aos modelos considerados, eles podem representar uma linha de transmissão com algumas restrições, por uma cascata de circuitos π. Cada circuito π é composto por uma resistência e indutância em série seguida por um paralelo de condutância e capacitância no caso do modelo inicial e é representado pelo mesmo esquema acrescentando uma resistência em paralelo a resistência e indutância no caso do circuito com resistência de amortecimento. As equações de estado são usadas para representar o sistema descrito, elas são obtidas usando tensões de capacitor e correntes de indutor como variáveis de estado. Os resultados geram um conjunto de equações de diferenças lineares usando integração trapezoidal. As equações resultantes são então aplicadas ao software computacional. Em seguida, as otimizações são aplicadas para a diminuição de tempo de processamento e memória. Essas otimizações são baseadas em técnicas de matrizes esparsas e decomposição LU (*lower and upper*).

Palavras-chave: análise numérica, linha de transmissão, otimização, simulação no domínio do tempo, fenômenos transitórios em sistemas de potência.

ABSTRACT

This work is based on the development and analysis of optimizations of two numerical models that represent transmission lines. The optimization has the idea of reducing the processing memory used in these models and also reducing their processing time. The analyzes presented in this work are not related to the minimization of numerical oscillations (Gibbs oscillations). In relation to the models considered, they can represent a transmission line with some restrictions, by a cascade of π circuits. Each π circuit is composed of a series resistance and inductance followed by a conductance and capacitance parallel in the case of the initial model and is represented by the same scheme adding a resistance in parallel to the resistance and inductance in the case of the circuit with damping resistance. The state equations are used to represent the described system, they are obtained using capacitor voltages and inductor currents as state variables. The results generate a set of linear difference equations using trapezoidal integration. The resulting equations are then applied to the computational software. Then, optimizations are applied to decrease processing time and memory. These optimizations are based on sparse matrix techniques and LU (lower and upper) decomposition.

Keywords: numerical analysis, transmission line, optimization, time-domain simulation, transient phenomena in power systems.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS, SIGLAS E ACRÔNIMOS

LU	Lower and Upper
SPS	Sparse
EMTP	Eletromagnetic Transients Program
ATP	Alternative Transient Program
R_D	Damping Resistance
PSCAD	Power System Computer Aided Design
R	Resistência (dada em Ohm, Ω)
L	Indutância (dada em Henry, H)
C	Capacitância (dada em Farad, F)
G	Condutância (dada em Ohm, Ω)
R'	Resistência por unidade de comprimento (Ω/km)
L'	Indutância por unidade de comprimento (H/km)
G'	Condutância por unidade de comprimento (Ω/km)
C'	Capacitância por unidade de comprimento (F/km)
n	Quantidade de circuitos π na cascata
Z	Impedância elétrica (dada em Ohm, Ω)
X_L	Reatância indutiva (dada em Ohm, Ω)
Ω	Unidade de medida de resistência
EMTDC	Electro-Magnetic Transient Design and Control

SUMÁRIO

1. INTRODUÇÃO

Este trabalho está relacionado a fenômenos transitórios eletromagnéticos em redes elétricas com enfoque principal em linhas de transmissão, tanto para transmissão de potência como para transmissão de sinal. A teoria de linhas de transmissão é aplicada tanto em sistemas de fornecimento de energia, que trabalham em baixa frequência, como em transmissão de sinais de dados, que é realizada em alta frequência. Com isso, este trabalho visa otimizar os modelos numéricos utilizados nas rotinas utilizadas nos dois tipos de sistemas de transmissão.

Linhas de transmissão são sistemas utilizados para transmitir energia eletromagnética entre dois nós diferentes de um sistema elétrico. Estão relacionadas a transmissão de potência em alta tensão até transmissão de dados em alta frequência. Essa transmissão é guiada de uma fonte geradora para uma carga consumidora utilizando fios torcidos e paralelos, cabos coaxiais ou guias de onda. Para fins de estudo, a modelagem dessas linhas de transmissão pode ser feita de forma relativamente simples, convertendo os parâmetros distribuídos do sistema estudado em parâmetros concentrados. Os modelos de circuitos elétricos mais utilizados neste tipo de análise são o circuito π e o circuito T.[1]

Nas rotinas utilizadas neste trabalho, a representação da linha de transmissão foi feita por meio de um circuito monofásico constituído por cascatas de circuitos π modificados. Essa cascata monofásica pode ser associada a linhas trifásicas considerando sistemas trifásicos equilibrados e balanceados, sendo que a análise é direcionada para uma das fases ou realizada tomando como base os valores de sequência positiva da modelagem em componentes simétricos.[2]

Usando a representação de linhas de transmissão como uma resistência e indutância em série seguidas por uma condutância em paralelo com uma capacitância, o modelo pode ser formulado usando as tensões no capacitor e correntes no indutor como as variáveis de estado. Aplicando integração trapezoidal, as equações de estado são transformadas em uma série de equações de diferenças lineares. Assim, a proposta é, a partir de simulações computacionais, analisar o comportamento de uma linha de transmissão ao ser submetida a transitórios eletromagnéticos e propor otimizações pertinentes a qualidade da rotina em si.[3]

Transitórios eletromagnéticos são fenômenos que ocorrem durante o regime transitório de um sistema elétrico e, no caso de linhas de transmissão, retiram tais circuitos de sua situação de funcionamento com valores de tensão, corrente e potência nominais e estáveis, podendo causar picos e oscilações de tensão, corrente ou potência acima da capacidade nominal dos sistemas mencionados.[4] Como os fenômenos transitórios podem trazer muitos prejuízos, concessionárias de energia e de transmissão de dados estão cada vez mais preocupadas com essas oscilações. Por isso, há a proposta de estudo sobre os efeitos que podem ser causados por essas situações. Com a aplicação de recursos computacionais, métodos numéricos foram implementados e diversos fenômenos puderam ser simulados por meio de modelos digitais. Com isso, a proposta de otimizar as rotinas que fazem uso desses métodos numéricos também se mostra algo válido.[5]

Grande parte dos programas disponíveis atualmente fazem uso de técnicas de modelagem numérica. Os principais deles que são utilizados para análises e simulações de fenômenos em linhas de transmissão empregam a integração trapezoidal, considerando o tempo como variável independente. Tais programas podem ser classificados como do tipo EMTP (MICROTRAN, ATP, PSCAD/EMTDC).[6]

Programas do tipo EMTP (*Electromagnetic Transient Programs*) são de utilização específica e apresentam custo elevado. Outro inconveniente desses programas é que os mesmos limitam a quantidade de circuitos π que podem ser utilizados para representar a linha. Desse modo, dependendo do comprimento da linha a ser representada, a qualidade dos resultados obtidos a partir das simulações pode ser comprometida. Por conta disso, uma alternativa é descrever as correntes e tensões na cascata de circuitos π por meio de variáveis de estado. As equações de estado são, então, transformadas em equações de diferenças e podem ser resolvidas utilizando qualquer linguagem computacional. As equações de estado, que são as tensões e correntes ao longo da linha, serão analisadas numericamente no ambiente MatLab™, pois o *software* permite estender os limites impostos pelos programas do tipo EMTP.[7-10]

Dessa forma, o software MatLab™, voltado para análises matemáticas matriciais e aplicações específicas em Engenharia, foi considerado como uma boa opção para o trabalho. Essa escolha favorece o desenvolvimento do projeto nas questões de flexibilidade para manipular matrizes e na facilidade para obtenção de gráficos. No caso específico do trabalho desenvolvido, são aplicadas técnicas de matrizes esparsas e decomposição LU em duas rotinas numéricas simples utilizadas para simulações de fenômenos transitórios eletromagnéticos em linhas de transmissão. O interesse principal é analisar se é possível diminuir significativamente o tempo computacional e a alocação de memória para a realização de simulação dos fenômenos mencionados. A utilização da esparsidade nas matrizes se faz interessante pois ela reduz os termos reservados de casa matriz dentro de uma rotina, fazendo com que o seu custo em memória seja consideravelmente menor e a decomposição LU tem o papel de substituir o cálculo de uma matriz inversa.

2. REVISÃO TEÓRICA

2.1 MODELO NUMÉRICO INICIAL

Para a modelagem com circuitos π, foi necessário o estudo da representação dos parâmetros distribuídos presentes em uma linha de transmissão real e sua respectiva conversão em parâmetros concentrados. A determinação precisa dos valores distribuídos presentes na linha deve ser baseada nas grandezas geométricas das torres da linha de transmissão e na resistência de terra da região por onde passa a linha.[11-12]

As equações de estado podem ser descritas por meio de um sistema linear como mostrado a seguir:

$$\dot{x} = [A]x + [B]u \tag{1}$$

Onde x é o vetor de variáveis de estado, u é vetor de entrada de dados, A é a matriz que representa o sistema e B é o vetor que introduz as fontes independentes do circuito analisado. É possível resolver (1) aplicando a integração trapezoidal, que é um procedimento numérico amplamente aplicado na resolução de equações diferenciais.

A integração trapezoidal é um método de integração numérica adequado para parâmetros discretos e consiste em uma forma aperfeiçoada do método da tangente ou mais conhecido como método de Euler. Geralmente, é aplicado às equações diferenciais de difícil resolução analítica.[13-14]

A integração trapezoidal aplicada em (1) leva ao seguinte resultado:

$$x_{k+1} - x_k = \frac{T}{2}\left(Ax_{k+1} + Bu_{k+1} + Ax_k + Bu_k\right) \tag{2}$$

Em que T é o passo de integração utilizado para resolver o sistema. Rearranjando (2), tem-se:

$$x_{k+1} = x_k + \frac{T}{2}\left(Ax_{k+1} + Bu_{k+1} + Ax_k + Bu_k\right) \tag{3}$$

Resolvendo (3), o sistema é reescrito como:

$$\left[I - \frac{T}{2}A\right]x_{k+1} = \left[I + \frac{T}{2}A\right]x_k + \frac{T}{2}B\left[u_k + u_{k+1}\right] \tag{4}$$

Em (4), I é a matriz identidade de ordem 2n, onde n é o número de circuitos π. Agrupando os termos em (4) obtemos (5):

$$x_{k+1} = A''x_k + B'\left[u_k + u_{k+1}\right] \tag{5}$$

Onde A', A'' e B' são matrizes constantes descritas por:

$$A' = \left[I - \frac{T}{2}A\right]^{-1};\ A'' = A' \cdot \left[I + \frac{T}{2}A\right];\ B' = A' \cdot \frac{T}{2}B \tag{6}$$

Os parâmetros R e L são resistência e indutância longitudinais da linha, respectivamente. Os parâmetros G e C são condutância e capacitância transversais de linha, respectivamente. Em (7), os elementos de cada circuito π são descritos em relação aos parâmetros por unidade de comprimento da linha (R', L', G' e C'). O parâmetro d é o comprimento da linha e n é o número de circuitos π utilizados para representar a linha de transmissão analisada.

$$R = R'\frac{d}{n};\quad L = L'\frac{d}{n};\quad G = G'\frac{d}{n}\quad C = C'\frac{d}{n} \tag{7}$$

A representação da linha de transmissão é mostrada na Figura 1.

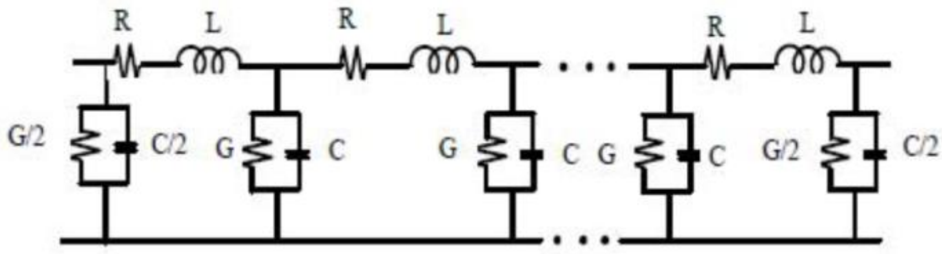

Figura 1 - Linha representada por meio de uma cascata de circuitos π

Fonte: Autor

A análise matemática da cascata de circuitos π, foi realizada com auxílio das leis de Kirchhoff. Vale lembrar que nessa cascata há três tipos de circuitos π, que apresentam detalhes diferentes na análise numérica devido à sua posição na cascata. São eles: o primeiro circuito, que é conectado às fontes no início da linha; os intermediários; e o último, que está em aberto.

O esquema do primeiro circuito π pode ser visto na Figura 2 e as expressões obtidas após análise de tal se apresentam em (8) e (9).

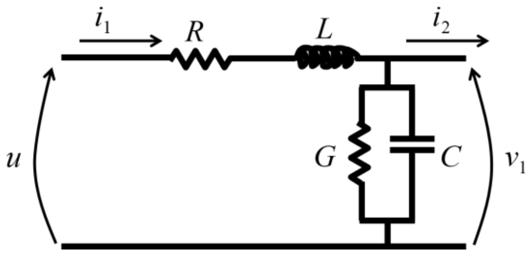

Figura 2 - Primeiro circuito π da cascata

Fonte: Autor

Fazendo análise de malha, obtém-se:

$$i'_1 = \frac{1}{L}u - \frac{R}{L}i_1 - \frac{1}{L}v_1 \tag{8}$$

Fazendo análise no nó, obtém-se:

$$v'_1 = \frac{1}{C}i_1 - \frac{G}{C}v_1 - \frac{1}{C}i_2 \tag{9}$$

O esquema do circuito π intermediário pode ser visto na Figura 3 e as expressões são mostradas em (10) e (11).

Fazendo análise de malha, obtém-se:

$$i'_j = \frac{1}{L}v_{j-1} - \frac{R}{L}i_j - \frac{1}{L}v_j \tag{10}$$

Fazendo análise no nó, obtém-se:

$$v'_j = \frac{1}{C}i_j - \frac{G}{C}v_j - \frac{1}{C}i_{j+1} \tag{11}$$

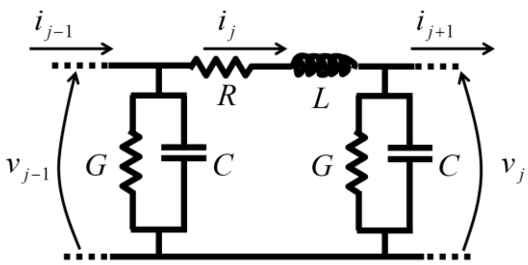

Figura 3 - Circuito π intermediário da cascata

Fonte: Autor

O esquema do último circuito π pode ser visto na Figura 4 e as expressões obtidas são mostradas em (12) e (13).

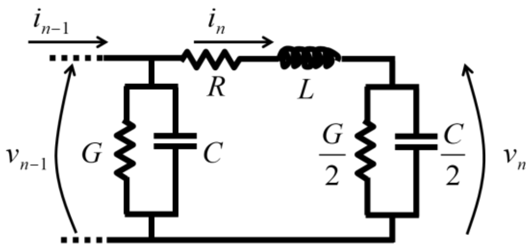

Figura 4 - Último circuito π da cascata

Fonte: Autor

Fazendo análise de malha, obtém-se:

$$i'_n = \frac{1}{L} v_{n-1} - \frac{R}{L} i_n - \frac{1}{L} v_n \tag{12}$$

Fazendo análise no nó, obtém-se:

$$v'_n = \frac{2}{C} i_n - \frac{G}{C} v_n \tag{13}$$

Utilizando as equações de estado obtidas acima, a estrutura da matriz A para representar uma linha de transmissão sem considerar a influência da frequência é mostrada em (16) e a estrutura do vetor B é mostrada em (14). O vetor de variáveis de estado é mostrado em (15). Nesse caso, foi considerado que a linha é alimentada por uma fonte de tensão ideal e o terminal de recepção está em aberto.

$$B = \begin{bmatrix} 1/L & 0 & \cdots & \cdots & 0 \end{bmatrix}^T \tag{14}$$

$$x = \begin{bmatrix} i_1 & v_1 & \cdots & i_2 & v_2 & \cdots & i_n & v_n \end{bmatrix}^T \tag{15}$$

$$A = \begin{bmatrix} -R/L & -1/L & 0 & \cdots & \cdots & 0 \\ 1/C & -G/C & -1/C & \ddots & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & 1/C & -G/C & -1/C & 0 \\ \vdots & \ddots & \ddots & 1/L & -R/L & -1/L \\ 0 & \cdots & \cdots & 0 & 2/C & -G/C \end{bmatrix} \tag{16}$$

2.2 MODELO NUMÉRICO COM RESISTÊNCIA DE AMORTECIMENTO

A segunda rotina inserida neste trabalho será a que se utilizada de um modelo numérico parecido com a rotina base, porém há o acréscimo de uma resistência de amortecimento (R_D). As equações de estado para o modelo numérico com resistência de amortecimento também podem ser descritas por meio de um sistema linear exatamente igual da modelo anterior (1).

A análise matemática da cascata de circuitos π após a introdução de R_D, que matematicamente é apresentado em (17), foi realizada com auxílio das leis de Kirchhoff. Vale lembrar que nessa cascata modificada há três tipos de circuitos π se comportam de maneira diferente. São eles: o primeiro circuito, que é conectado às fontes no início da linha; os intermediários; e o último, que representa o terminal de carga da linha em aberto.

$$R_D = r_d * \frac{2L}{\Delta t} \; ; \; G_D = \frac{\Delta t}{2r_d * L} \tag{17}$$

O esquema do primeiro circuito π pode ser visto na Figura 5 e as expressões obtidas mostradas em (18) e (19).

Fazendo análise de malha, obtém-se:

$$i'_1 = \frac{1}{L}u - \frac{R}{L}i_1 - \frac{1}{L}v_1 \tag{18}$$

Fazendo análise no nó, obtém-se:

$$v'_1 = \frac{1}{C}i_1 - \frac{(2G_D+G)}{C}v_1 + \frac{G_D}{C}(u+v_2) - \frac{1}{C}i_2 \tag{19}$$

Sendo G_D a condutância de amortecimento, ou seja, o inverso de R_D.

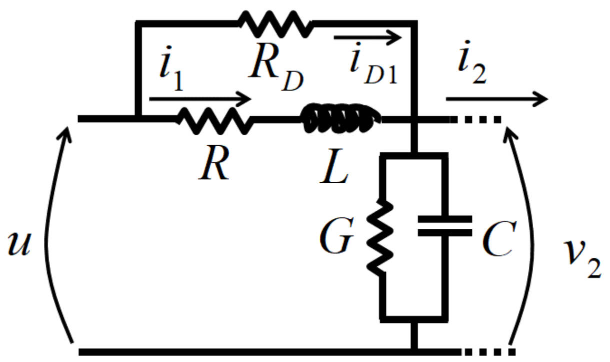

Figura 5 - Primeiro circuito π da cascata modificada

Fonte: Autor

O esquema do circuito π intermediário pode ser visto na Figura 6 e as expressões obtidas são mostradas em (20) e (21).

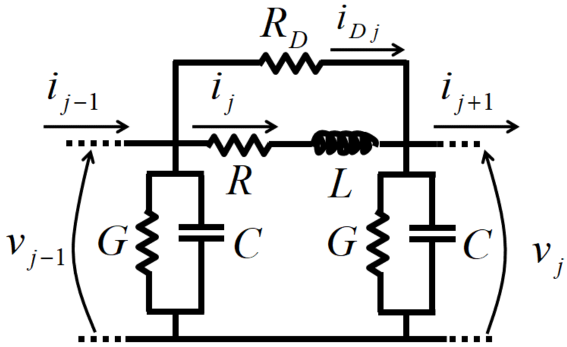

Figura 6 - Circuito π intermediário da cascata modificada

Fonte: Autor

Fazendo análise de malha, obtém-se:

$$i'_j = \frac{1}{L} v_{j-1} - \frac{R}{L} i_j - \frac{1}{L} v_j \tag{20}$$

Fazendo análise no nó, obtém-se:

$$v'_j = \frac{1}{C} i_j - \frac{(2G_D + G)}{C} v_j + \frac{G_D}{C} (v_{j-1} + v_{j+1}) - \frac{1}{C} i_{j+1} \tag{21}$$

O esquema do último circuito π pode ser visto na Figura 7 e as expressões obtidas são mostradas em (22) e (23).

Fazendo análise de malha, obtém-se:

$$i'_n = \frac{1}{L} v_{1-1} - \frac{R}{L} i_n - \frac{1}{L} v_n \tag{22}$$

Fazendo análise no nó, obtém-se:

$$v'_n = \frac{2}{C} i_n + \frac{2G_D}{C} (v_{n-1}) - \frac{(2G_D + G)}{C} v_n \tag{23}$$

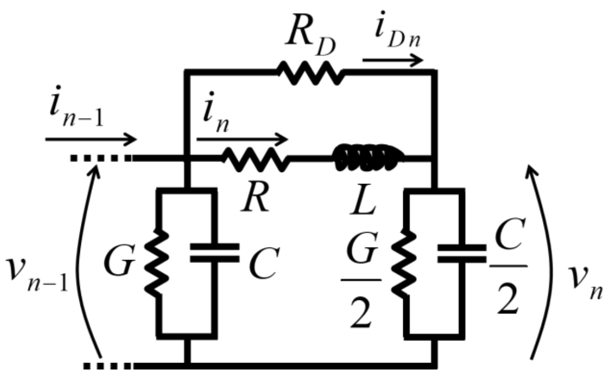

Figura 7 - Último circuito π da cascata modificada

Fonte: Autor

Utilizando as equações obtidas juntamente com o conceito de equações de estados descrito acima, a estrutura da matriz A foi determinada e é apresentada em (26). Os elementos apresentados na matriz A são definidos em (24) e (25). Os vetores B e x deste modelo numérico foram definidos em (14) e (15).

$$R_D = k\frac{2L}{\Delta t} \tag{24}$$

$$G_D = \frac{1}{k}\frac{\Delta t}{2L} \tag{25}$$

$$A = \begin{bmatrix} -\frac{R}{L} & -\frac{1}{L} & 0 & 0 & 0 & 0 & \cdots & \cdots & 0 \\ \frac{1}{C} & -\frac{2G_D-G}{C} & -\frac{1}{C} & \frac{G_D}{C} & 0 & 0 & \cdots & \cdots & \vdots \\ 0 & \frac{1}{L} & -\frac{R}{L} & -\frac{1}{L} & 0 & 0 & \cdots & \cdots & \vdots \\ 0 & \frac{G_D}{C} & \frac{1}{C} & -\frac{2G_D-G}{C} & -\frac{1}{C} & \frac{G_D}{C} & \cdots & \cdots & \vdots \\ \vdots & \cdots & \cdots & \frac{1}{L} & -\frac{R}{L} & -\frac{1}{L} & \cdots & \cdots & \vdots \\ \vdots & \cdots & \cdots & \frac{G_D}{C} & \frac{1}{C} & -\frac{2G_D-G}{C} & -\frac{1}{C} & \frac{G_D}{C} & \vdots \\ \vdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & 0 \\ \vdots & \cdots & \cdots & \cdots & \cdots & 0 & \frac{1}{L} & -\frac{R}{L} & -\frac{1}{L} \\ 0 & \cdots & \cdots & \cdots & \cdots & 0 & \frac{2G_D}{C} & \frac{2}{C} & -\frac{2G_D-G}{C} \end{bmatrix} \tag{26}$$

Por meio de (30), (31) e (32), uma constante de amortecimento *k* é introduzida na modelagem de circuitos π, resultando na redução de erros causados por oscilações numéricas.

2.3 TÉCNICAS AUXILIARES PARA AS OTIMIZAÇÕES

As duas técnicas que foram utilizadas para gerar otimizações nas rotinas numéricas foram a esparsidade e a decomposição LU. Essas técnicas auxiliam com relação a utilização de procedimentos matemáticos com matrizes.

Por definição, matrizes esparsas são matrizes nas quais grande parte de seus elementos têm valor nulo. Matrizes esparsas têm características contrárias às matrizes densas. Nas matrizes densas, a maioria dos elementos não são nulos. Para se definir uma matriz como esparsa, a relação entre a quantidade de elementos nulos e a quantidade total de elementos é calculada. Esse valor é definido como fator de dispersão da matriz e é por meio desse fator que as matrizes são caracterizadas como densas ou esparsas. Se esse fator for maior que 0,5, a matriz é considerada esparsa.[16]

O *software* MatLab™, que é utilizado para a execução das rotinas que serão otimizadas neste trabalho, possui um comando chamado *sparse*, o qual é amplamente utilizado em análises e cálculos envolvendo matrizes esparsas.[17] Tal rotina tem o propósito de armazenar apenas valores não nulos, fazendo com que o cálculo realizado com matrizes esparsas utilize uma menor quantidade de memória para seu processamento, diminuindo também o tempo computacional necessário para o cálculo mencionado.

Em relação as rotinas utilizadas para as simulações de transitórios eletromagnéticos, na matriz identificada como A, os valores não nulos estão dispostos apenas na diagonal principal e nas diagonais acima e abaixo subsequentes a ela. Essa estrutura é mostrada na Figura 8. Por conta de representação da matriz A e também do uso de uma ferramenta do *software* MatLab™ chamada *issparse*, que ao ser utilizada retorna o valor lógico 1 se a matriz for realmente esparsa, pode-se determinar a matriz A como uma matriz esparsa e que poderia receber as otimizações.

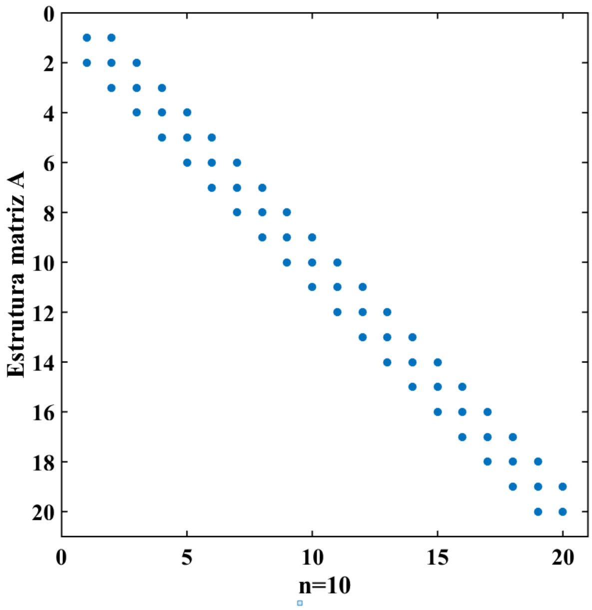

Figura 8 - Representação da matriz A.

Fonte: Autor

A decomposição LU (*lower and upper*) tem como procedimento a fatoração de uma matriz não singular, que é a matriz inicial, como o produto de uma matriz triangular inferior (lower) e uma matriz triangular superior (upper). Esta decomposição é utilizada em análises numéricas para resolver sistemas de equações mais eficientes e também para encontrar matrizes inversas. Neste trabalho, a utilização da decomposição LU será no cálculo mais efetivo de uma matriz inversa. Na Figura 9 é apresentado o processo de decomposição de uma matriz qualquer.[15]

$$\begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ l_{21} & 1 & 0 \\ l_{31} & l_{32} & 1 \end{pmatrix} \begin{pmatrix} u_{11} & u_{12} & u_{13} \\ 0 & u_{22} & u_{23} \\ 0 & 0 & u_{33} \end{pmatrix}$$

Figura 9 - Representação da decomposição LU

Fonte: Autor

3. DESENVOLVIMENTO E RESULTADOS

3.1 OTIMIZAÇÃO BASEADA EM MATRIZES ESPARSAS

O circuito utilizado é uma cascata de circuitos π que simula uma linha de transmissão com uma fonte de tensão como alimentação e com saída em aberto. Antes de iniciar qualquer processo de otimização, foi necessário definir na rotina quais das matrizes gerariam benefícios em relação a memória e tempo de processamento com a utilização das otimizações. A otimização da matriz A tem significativa importância. Na rotina numérica de simulação de transitórios, tal matriz tem a maior quantidade de termos nulos e, consequentemente, é a matriz que utiliza a maior alocação de memória durante o processamento. Por isso, será a matriz onde todas as otimizações foram focadas e a qual será apresentada neste trabalho.

A partir disso, para o início do estudo proposto das otimizações em relação à memória alocada, se fez necessário a análise inicial das rotinas sem nenhum tipo de otimização. Na Tabela 1, são apresentados os valores de memória alocada da rotina base em Mb (Megabytes) para valores de circuitos π.

Tabela 1 - Memória alocada da rotina base não otimizada

Fonte: Autor

Circuitos π	Não Otimizado (Mb)
50	0,080
100	0,320
200	1,280
400	5,120
600	11,520
800	20,480
1000	32,000
1200	46,080
1400	62,720
1600	81,920
1800	103,680
2000	128,000

A partir da Tabela 1 é possível visualizar que os valores de memória aumentam significativamente com relação ao aumento de circuitos π, fazendo com que em simulações mais precisas (com números elevados de circuitos π), a rotina seja extremamente pesada com relação à memória. Na Figura 10, é apresentada a memória alocada da rotina base por circuitos π, fazendo com que se possa ver os resultados da Tabela 1 graficamente.

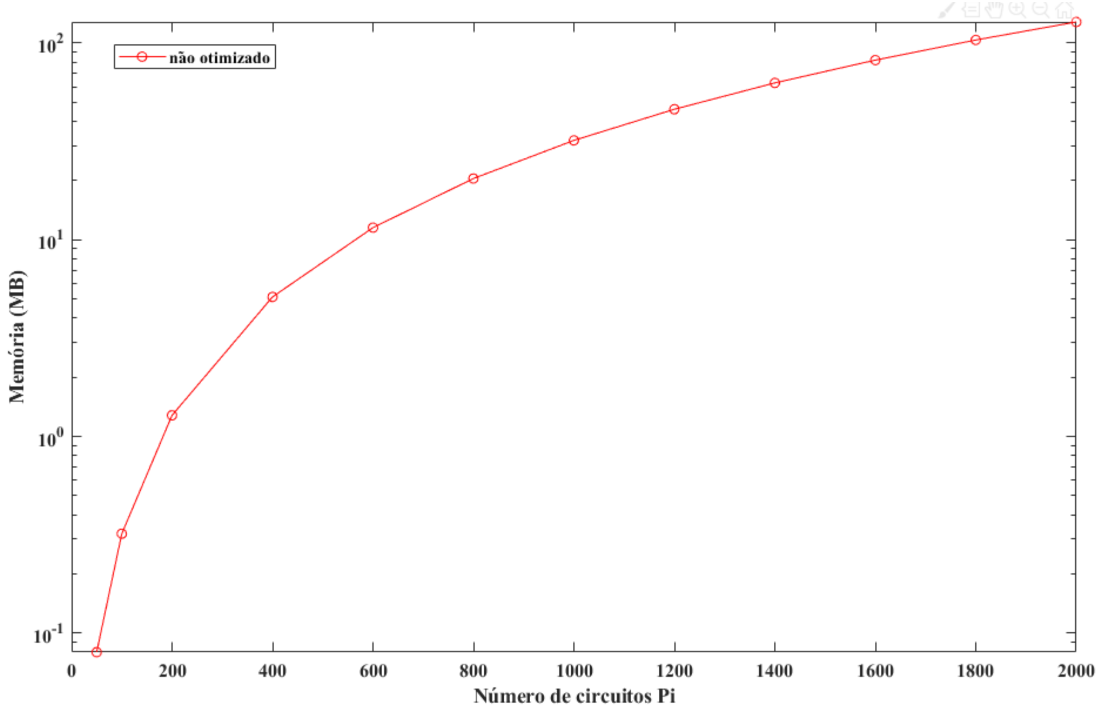

Figura 10 - Gráfico de memória alocada da rotina base não otimizada em escala logarítmica

Fonte: Autor

Assim como foi feito em relação à memória alocada da rotina base sem otimizações, também se faz necessário apresentar o tempo de processamento sem nenhum tipo de otimização para fins de comparação com a otimizações propostas neste trabalho.

Por conta disso, na Tabela 2, são apresentados os valores de tempo de processamento da rotina base em segundos para valores de circuitos π. A partir dos valores na Tabela 2, é possível visualizar que o tempo de processamento aumenta significativamente com relação ao aumento de circuitos π, fazendo com que em simulações com números elevados de circuitos π a rotina seja extremamente lenta. Na Figura 11, é apresentado o tempo de processamento da rotina base por circuitos π, possibilitando a visualização resultados da Tabela 2.

Tabela 2 - Tempo de processamento da rotina base não otimizada

Fonte: Autor

Circuitos π	Não Otimizado (s)
50	0,103
100	0,125
200	0,257
400	2,195
600	5,353
800	9,559
1000	13,946
1200	19,951
1400	28,433
1600	36,924
1800	49,371
2000	61,044

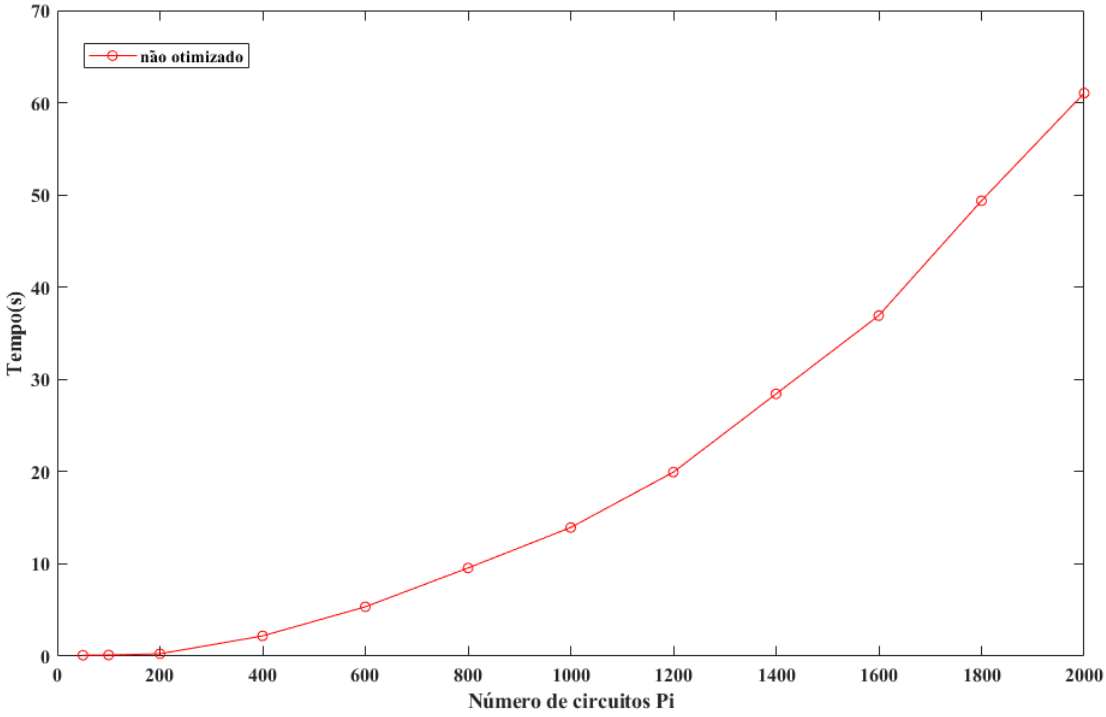

Figura 11 - Gráfico do tempo de processamento da rotina base não otimizada

Fonte: Autor

Seguindo a mesma estratégia, também se faz necessário fazer a apresentação dos dados de alocação de memória e de tempo de processamento para a rotina que integra a resistência de amortecimento (R_D). Na Tabela 3, são apresentados os valores de memória alocada da rotina R_D em Mb (Megabytes) para valores de circuitos π.

Na Tabela 3, é possível visualizar que os valores de memória aumentam exponencialmente com relação ao aumento de circuitos π, fazendo com que em simulações mais precisas (com números elevados de circuitos π) a rotina seja extremamente pesada com relação à memória. Na Figura 12, é apresentada a memória alocada da rotina base por circuitos π, fazendo com que se possa ver os resultados da Tabela 3 graficamente.

Tabela 3 - Memória alocada da rotina R_D não otimizada

Fonte: Autor

Circuitos π	Não Otimizado (Mb)
50	0,080
100	0,320
200	1,280
400	5,120
600	11,520
800	20,480
1000	32,000
1200	46,080
1400	62,720
1600	81,920
1800	103,680
2000	128,000

Da mesma forma que foi apresentada a memória alocada da rotina R_D sem otimizações, é necessário apresentar o tempo de processamento sem nenhum tipo de otimização para fins de comparação com a otimizações propostas nos tópicos demais tópicos. Com isso, dando sequência aos dados apresentados, na Tabela 4, são apresentados os valores de tempo de processamento da rotina R_D em segundos para valores de circuitos π.

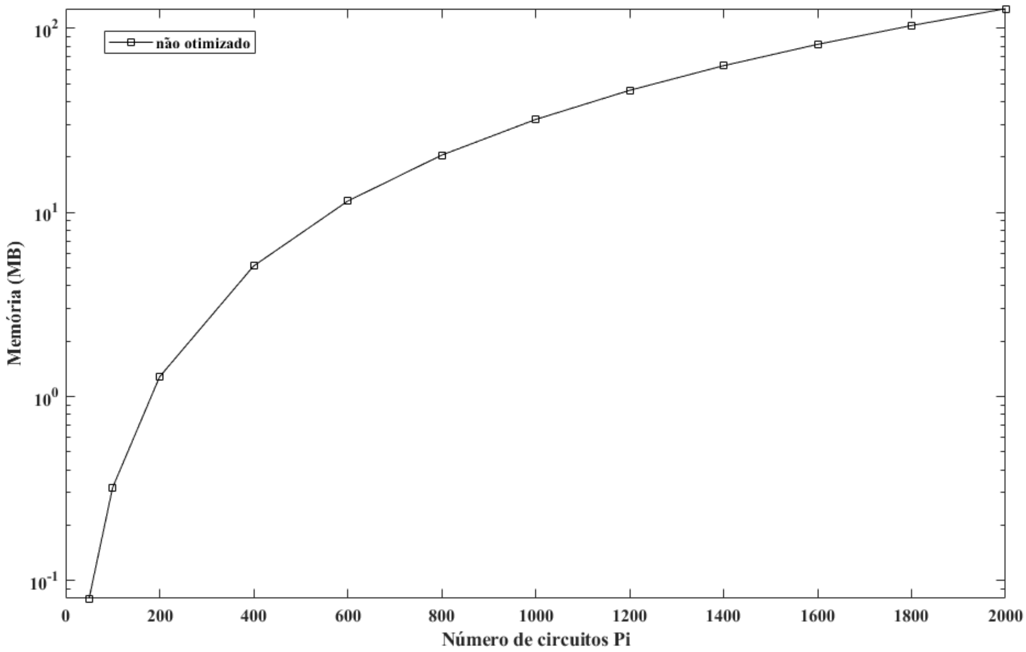

Figura 12 - Gráfico de memória alocada da rotina R_D não otimizada em escala logarítmica

Fonte: Autor

Tabela 4 - Tempo de processamento da rotina R_D não otimizada

Fonte: Autor

Circuitos π	Não Otimizado (s)
50	0,061
100	0,089
200	0,168
400	1,934
600	4,495
800	8,209
1000	12,561
1200	18,819
1400	26,619
1600	35,424
1800	46,760
2000	59,414

Com os dados fornecidos pela Tabela 4, foi possível gerar o gráfico apresentado na Figura 13. A partir deles é possível visualizar que o tempo de processamento aumenta exponencialmente com relação ao aumento de circuitos π, fazendo com que em simulações com números elevados de circuitos π, a rotina seja extremamente lenta chegando a valores altíssimos com uma quantidade de circuitos π maior que 2000.

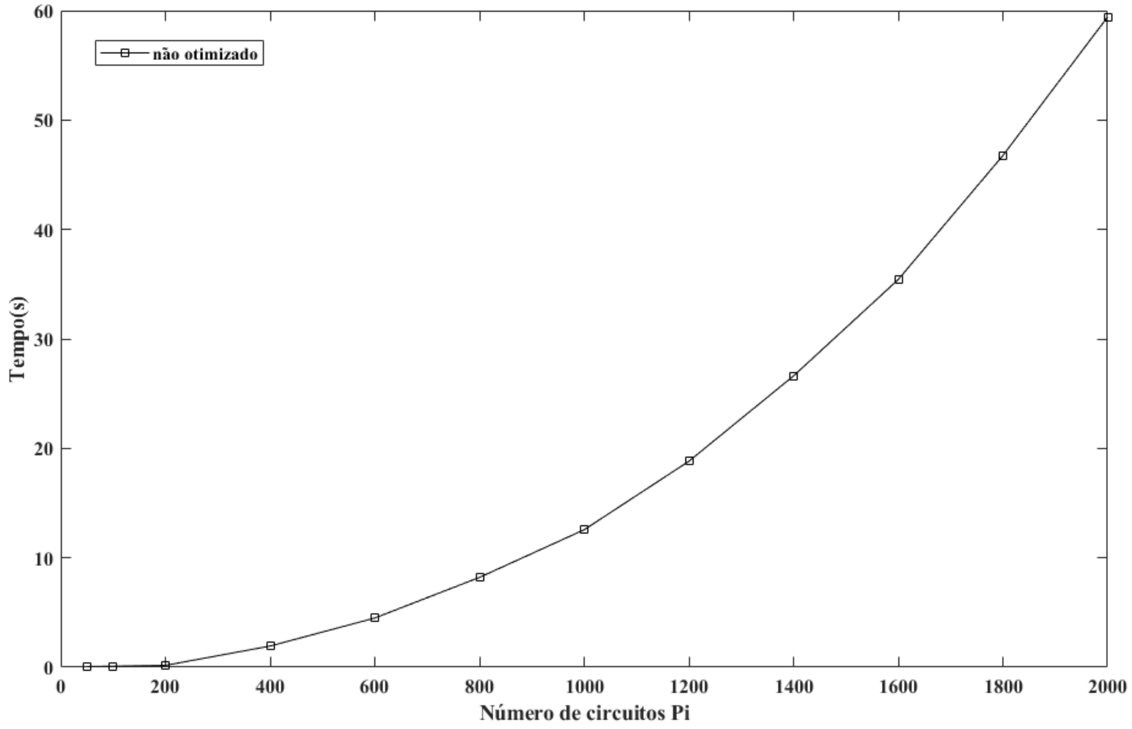

Figura 13 - Gráfico do tempo de processamento da rotina R_D não otimizada

Fonte: Autor

A primeira otimização proposta para este trabalho é utilizando da técnica de esparsidade. Para fazer o uso desta técnica na rotina base e também na rotina R_D, é necessário fazer uso da ferramenta *sparse* que o *softaware* MatLab™ fornece. Com esta ferramenta e com a definição da matriz A como a mais benéfica fazendo uso das técnicas de otimização nas rotinas, foi necessário substituir a definição da matriz A por *sparse*(A) para que a técnicas fossem implementadas nas rotinas.

A utilização da ferramenta *sparse* faz com que todos os termos nulos, ou seja, com o valor zero fossem armazenados, restando apenas os valores não nulos para que fossem efetuados os cálculos da rotina. A matriz A foi considerada a melhor para se utilizar esta técnica, pois apenas os termos laterais à diagonal principal e a própria diagonal principal possuem valores não nulos. Na tabela 5, são apresentados os valores de memória alocada da rotina base com a otimização SPS (por esparsidade) em comparação aos valores não otimizados.

Tabela 5 - Memória alocada da rotina base otimizada por esparsidade

Fonte: Autor

Circuitos π	Não Otimizado (Mb)	Otimização SPS (Mb)
50	0,080	0,005
100	0,320	0,011
200	1,280	0,022
400	5,120	0,045
600	11,520	0,067
800	20,480	0,090
1000	32,000	0,110
1200	46,080	0,130
1400	62,720	0,160
1600	81,920	0,190
1800	103,680	0,210
2000	128,000	0,230

A partir da Tabela 5, é possível visualizar que os valores de memória diminuíram drasticamente com a utilização da matriz esparsa em relação a matriz não otimizada. Na Figura 14, são apresentados todos os resultados da Tabela 5 graficamente. A partir da Tabela 5 e da Figura 14, observa-se a diminuição expressiva da memória alocada que a rotina base utiliza após aplicar a técnica de esparsidade na matriz A. Para valores baixos de circuitos π a diminuição já é algo a se considerar, porém, o resultado significativo está relacionado a valores altos de circuitos π. Por exemplo, o valor de 2000 circuitos, onde há diminuição de mais de 99,8%. Isso ocorre porque a dimensão da matriz A está diretamente ligada a quantidade de circuitos π inseridas na simulação. A matriz A é uma matriz quadrada de ordem 2n, onde n é o número de circuitos π. Como ela tem a maior parte de seus termos nulos, a esparsidade funciona muito bem, pois elimina todos eles e economiza a memória que seria designada para armazená-los.

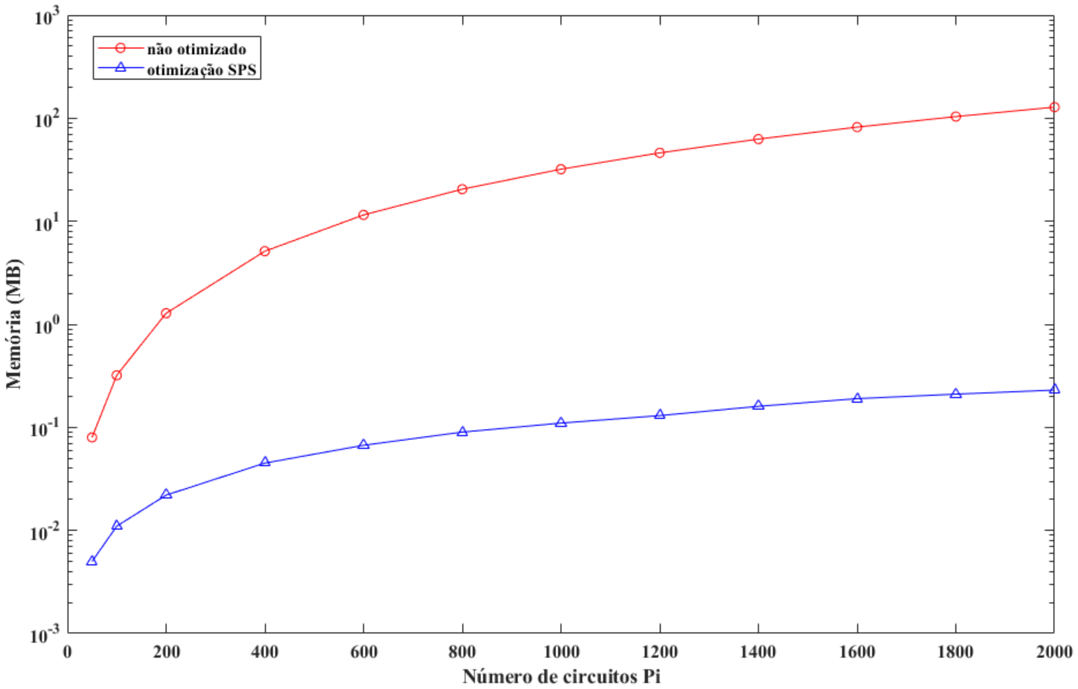

Figura 14 - Gráfico de memória alocada da rotina base otimizada por esparsidade em escala logarítmica

Fonte: Autor

Tabela 6 - Tempo de processamento da rotina base otimizada por esparsidade

Fonte: Autor

Circuitos π	Não Otimizado (s)	Otimização SPS (s)
50	0,103	0,077
100	0,125	0,087
200	0,257	0,148
400	2,195	0,503
600	5,353	2,838
800	9,559	5,186
1000	13,946	8,389
1200	19,951	11,721
1400	28,433	16,396
1600	36,924	21,959
1800	49,371	29,436
2000	61,044	37,851

Também se faz necessário apresentar o tempo de processamento. Por isso, na Tabela 6 e na Figura 15, são apresentados os valores de tempo de processamento, dados em segundos, da rotina base com a esparsidade.

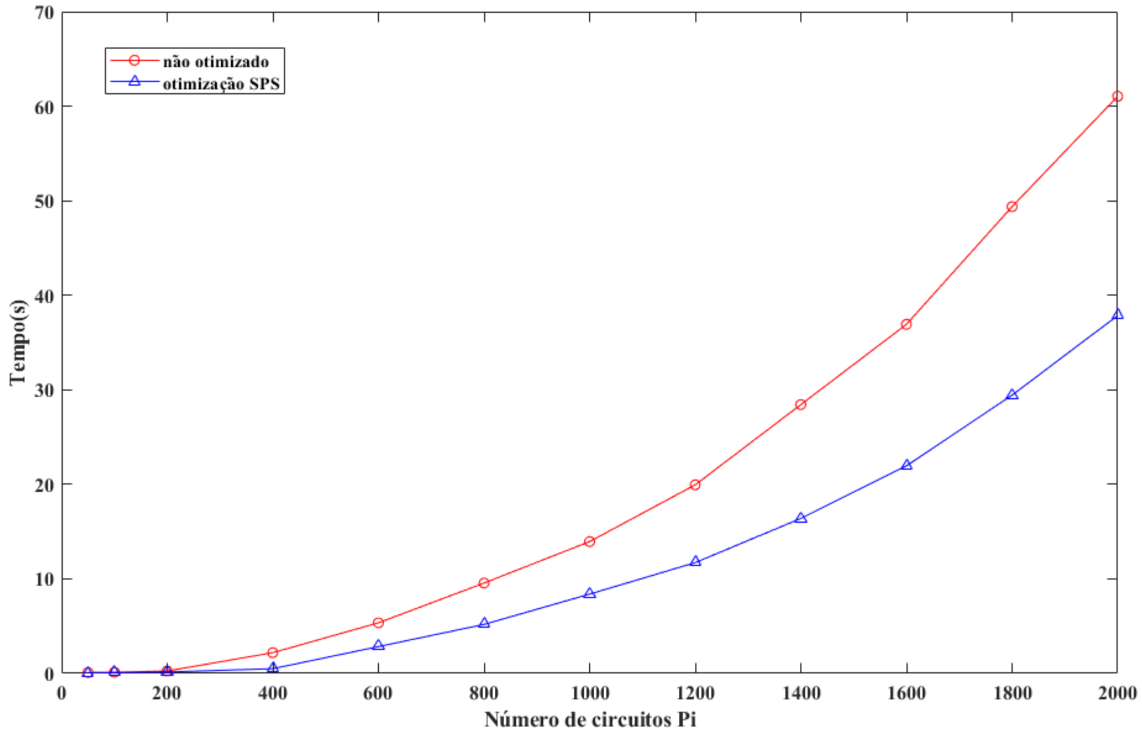

Figura 15 - Gráfico do tempo de processamento da rotina base otimizada por esparsidade

Fonte: Autor

A partir dos valores dados na Tabela 6 e na Figura 15, é possível visualizar que o tempo de processamento da rotina com a utilização da esparsidade diminui, em média, 40% para todas as quantidades de circuitos π. Isso ocorre, pois, a técnica de esparsidade não está diretamente ligada com o processo de cálculo da rotina, que tende a ser o processo mais longo. Por isso os resultados em questão de tempo de processamento são menos expressivos do que os de memória alocada.

Do mesmo jeito que foi tratada a rotina base, também se faz necessário tirar a apresentação dos dados de alocação de memória e de tempo de processamento para a rotina que integra a resistência de amortecimento (R_D). Com isso, na Tabela 7, são apresentados os valores de memória alocada da rotina R_D em Mb (Megabytes) para valores de circuitos π.

Tabela 7 - Memória alocada da rotina R_D otimizada por esparsidade

Fonte: Autor

Circuitos π	Não Otimizado (Mb)	Otimização SPS (Mb)
50	0,080	0,007
100	0,320	0,014
200	1,280	0,028
400	5,120	0,057
600	11,520	0,086
800	20,480	0,115
1000	32,000	0,143
1200	46,080	0,172
1400	62,720	0,201
1600	81,920	0,230
1800	103,680	0,259
2000	128,000	0,287

A partir dos resultados da Tabela 7, é possível visualizar que os valores de memória alocada da rotina R_D apresentaram valores baixíssimos com a utilização da técnica de esparsidade na matriz A em comparação aos resultados de memória alocada da rotina utilizando a matriz A não otimizada. Na Figura 16, são apresentados todos os resultados da Tabela 7 graficamente.

A partir da Tabela 7 e da Figura 16, observa-se diminuição significativa da memória alocada que a rotina R_D utiliza após aplicação da técnica de esparsidade na matriz A. Para valores baixos de circuitos π, a diminuição já é significativa, com valores em torno de 48%. Porém, o real resultado significativo é relacionado a valores altos de circuitos π, como por exemplo o valor de 2000 circuitos, onde há uma diminuição de mais de 99,7%. Isso ocorre, pois, a dimensão da matriz A está diretamente ligada a quantidade de circuitos π inseridas na simulação. A matriz A é uma matriz quadrada de ordem 2n, onde n é o número de circuitos π. Como ela tem a maior parte de seus termos nulos, a esparsidade funciona muito bem, pois elimina todos eles e economiza a memória que seria designada para armazená-los.

Também se faz necessário apresentar o tempo de processamento. Por isso, na Tabela 8 e na Figura 17, são apresentados os valores de tempo de processamento, dados em segundos, da rotina R_D com a esparsidade.

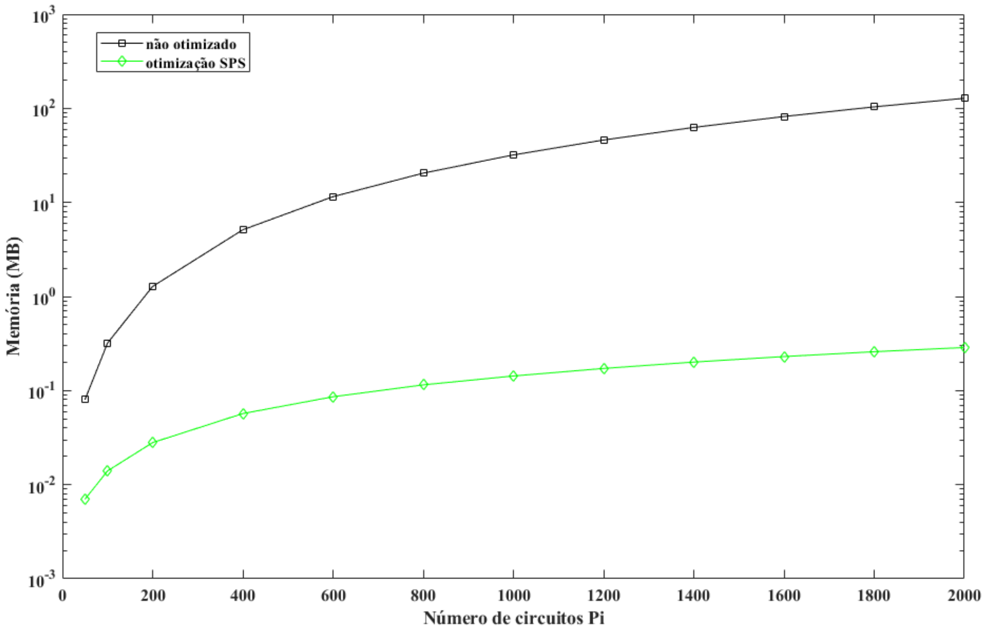

Figura 16 - Gráfico de memória alocada da rotina R_D otimizada por esparsidade em escala logarítmica

Fonte: Autor

Tabela 8 - Tempo de processamento da rotina R_D otimizada por esparsidade

Fonte: Autor

Circuitos π	Não Otimizado (s)	Otimização SPS (s)
50	0,061	0,076
100	0,089	0,084
200	0,168	0,139
400	1,934	0,643
600	4,495	2,660
800	8,209	5,174
1000	12,561	8,605
1200	18,819	12,829
1400	26,619	18,217
1600	35,424	25,125
1800	46,760	33,388
2000	59,414	43,256

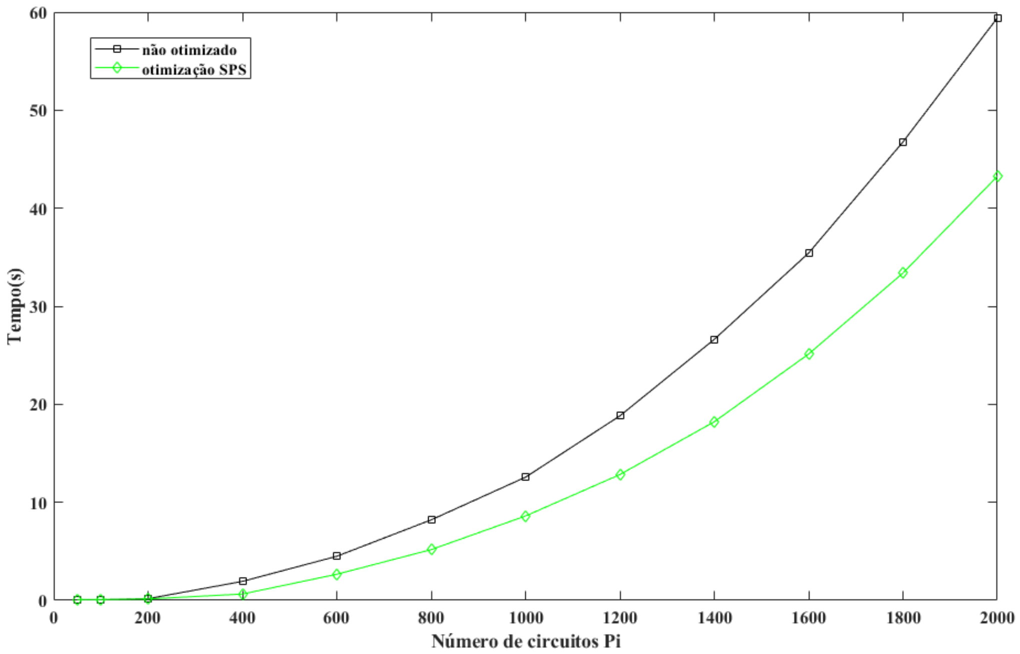

Figura 17 - Gráfico do tempo de processamento da rotina R_D otimizada por esparsidade

Fonte: Autor

A partir dos valores dados na Tabela 8 e na Figura 17, é possível visualizar que o tempo de processamento da rotina com a utilização da esparsidade diminui, em média 25%, para todas as quantidades de circuitos π. Isso ocorre, pois, a técnica de esparsidade não está diretamente ligada com o processo de cálculo da rotina, que tende a ser o processo mais longo, por isso os resultados em questão de tempo de processamento são menos expressivos do que os de memória alocada.

3.2 OTIMIZAÇÃO BASEADA EM DECOMPOSIÇÃO LU

Para a segunda otimização proposta no trabalho, será utilizada a técnica de decomposição LU. Esta técnica se incorpora nas rotinas substituindo a geração da matriz A' que era baseada em uma matriz inversa. A equação (27) mostra o passo da rotina que será substituído pela técnica de otimização.

$$A' = inv\left(Ident - \left(\left(\frac{dt}{2}\right) * A\right)\right) \tag{27}$$

Após a utilização da decomposição LU, (27) passa a ser (28) dentro das duas rotinas abordadas no trabalho.

$$[LL,UU] = lu(Ident - \left(\frac{dt}{2}\right) * A) \tag{28}$$

Para modificar a geração de umas das matrizes essenciais das rotinas, foi necessário fazer uma adaptação da equação final que era (29).

$$x = A' * (A" * x + B1 * du) \tag{29}$$

Após a inserção da otimização, (29) passa a ser (30) para que a decomposição feita anteriormente seja resgatada e calculado ao final do processo.

$$x = UU/(\frac{LL}{A" * x + B1 * du}) \tag{30}$$

Vale ressaltar que todas as modificações geradas pela inserção da decomposição LU são validas tanto para a rotina base quanto para a rotina com a resistência de amortecimento.

A ideia em se utilizar esta técnica de otimização é que o processo de geração de uma matriz inversa é muito custoso dentro da rotina tanto em relação a memória quanto em relação ao tempo. Ao retirar esse processo, pode-se ter um ganho em ambas as rotinas.

Nas figuras e tabelas deste capítulo serão apresentados os resultados desta otimização para ambas as rotinas com relação a memória alocada e tempo de processamento. Na apresentação destes resultados, eles também estarão sendo comparados com a otimização apresentada no tópico anterior e com a rotina sem nenhuma otimização. Para dar início, na Tabela 9 e na Figura 18, são apresentados os valores de memória alocada da rotina base com a decomposição LU comparados aos resultados das técnicas anteriores.

A partir da Tabela 9 e da Figura 18, percebe-se que a técnica de decomposição LU não traz nenhum benefício com relação a alocação de memória da rotina, fazendo com que os valores da memória alocada sejam iguais aos valores da rotina sem otimização e maiores em relação a otimização por esparsidade. Isso ocorre, pois, o papel da decomposição LU na rotina é substituir o cálculo da matriz inversa que é muito custoso em relação ao tempo de processamento da rotina. Esta técnica não traz nenhuma melhoria com relação à memória alocada.

Também se faz necessário apresentar o tempo de processamento. Por isso, na Tabela 10, são apresentados os valores de tempo de processamento, dados em segundos, da rotina base com a decomposição LU.

Tabela 9 - Memória alocada da rotina base otimizada por decomposição LU

Fonte: Autor

Circuitos π	Não Otimizado (Mb)	Otimização SPS (Mb)	Otimização LU (Mb)
50	0,080	0,005	0,080
100	0,320	0,011	0,320
200	1,280	0,022	1,280
400	5,120	0,045	5,120
600	11,520	0,067	11,520
800	20,480	0,090	20,480
1000	32,000	0,110	32,000
1200	46,080	0,130	46,080
1400	62,720	0,160	62,720
1600	81,920	0,190	81,920
1800	103,680	0,210	103,680
2000	128,000	0,230	128,000

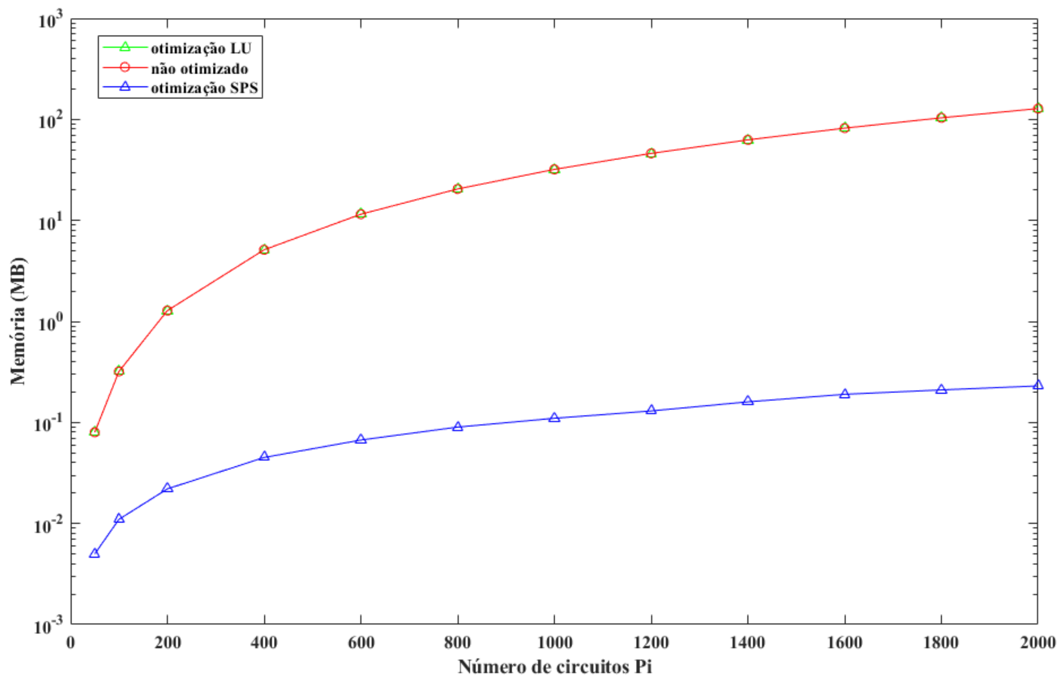

Figura 18 - Gráfico de memória alocada da rotina base otimizada por decomposição LU em escala logarítmica

Fonte: Autor

Tabela 10 – Tempo de processamento da rotina base otimizada por decomposição LU

Fonte: Autor

Circuitos π	Não Otimizado (s)	Otimização SPS (s)	Otimização LU (s)
50	0,103	0,077	0,200
100	0,125	0,087	0,460
200	0,257	0,148	1,422
400	2,195	0,503	6,837
600	5,353	2,838	14,761
800	9,559	5,186	26,036
1000	13,946	8,389	40,786
1200	19,951	11,721	57,989
1400	28,433	16,396	78,889
1600	36,924	21,959	102,997
1800	49,371	29,436	132,071
2000	61,044	37,851	164,058

A partir dos resultados da Tabela 10, é possível visualizar que os valores de tempo de processamento da rotina base são extremamente altos comparados com os resultados da rotina sem otimização e também com a otimização por esparsidade. Isso ocorre, pois, a decomposição LU divide a matriz calculada em duas novas matrizes, uma triangular superior e outra triangular inferior e a partir dessas duas matrizes a técnica calcula a matriz inversa. Os valores de tempos foram tão altos em comparação aos anteriores, pois, como grande parte dos termos da matriz A são nulos, ao se utilizar a decomposição LU, foi efetuado o cálculo com todos estes termos nulos que estavam alocados pela rotina. Isso não ocorria sem a decomposição LU. Na Figura 19, são apresentados todos os resultados da Tabela 10 graficamente.

Para a rotina R_D também foi utilizada a decomposição LU para testar sua eficácia. Na Tabela 11 e na Figura 20 são apresentados os resultados de memória alocada da rotina R_D com a decomposição LU, comparados aos resultados das técnicas anteriores.

A partir da Tabela 11 e da Figura 20, percebe-se que, a técnica de decomposição LU não traz nenhum benefício com relação a alocação de memória na rotina R_D. A conclusão é que o papel da decomposição LU na rotina é substituir o cálculo da matriz inversa que é muito custoso em relação ao tempo de processamento da rotina, por isso esta técnica não traz nenhuma melhoria.

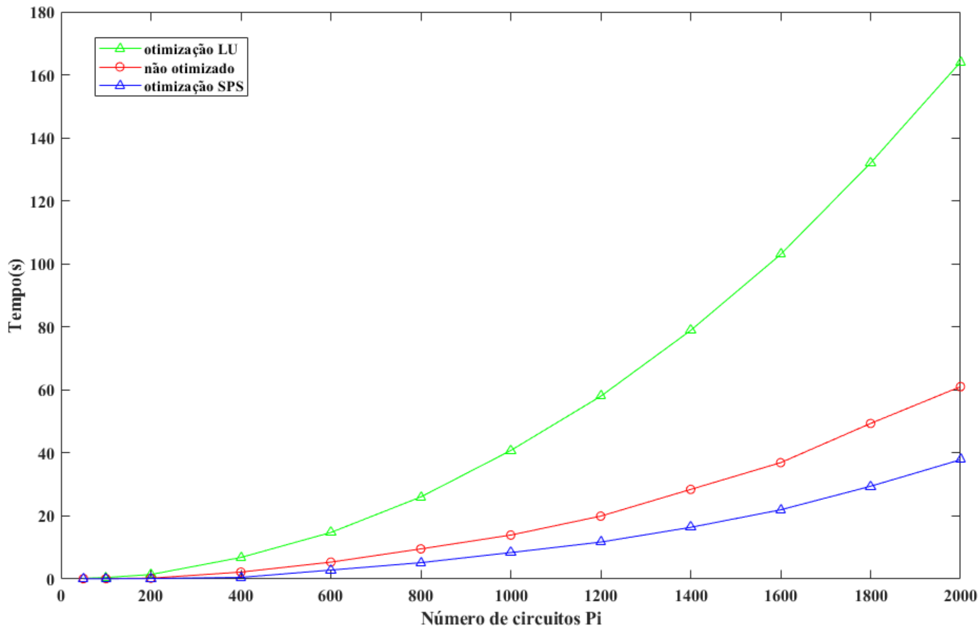

Figura 19 - Gráfico do tempo de processamento da rotina base otimizada por decomposição LU

Fonte: Autor

Tabela 11 - Memória alocada da rotina R_D otimizada por decomposição LU

Fonte: Autor

Circuitos π	Não Otimizado (Mb)	Otimização SPS (Mb)	Otimização LU (Mb)
50	0,080	0,007	0,080
100	0,320	0,014	0,320
200	1,280	0,028	1,280
400	5,120	0,057	5,120
600	11,520	0,086	11,520
800	20,480	0,115	20,480
1000	32,000	0,143	32,000
1200	46,080	0,172	46,080
1400	62,720	0,201	62,720
1600	81,920	0,230	81,920
1800	103,680	0,259	103,680
2000	128,000	0,287	128,000

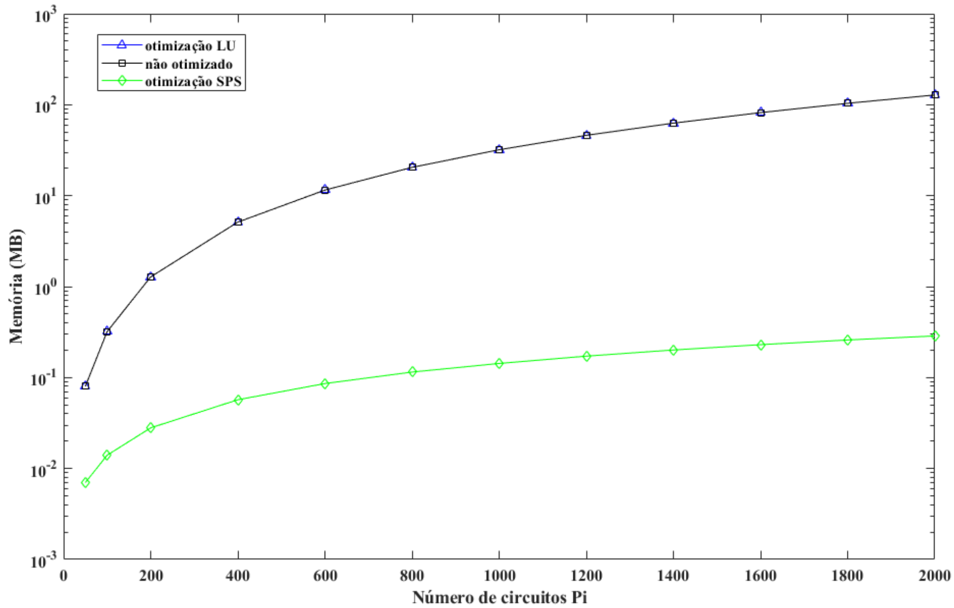

Figura 20 - Gráfico de memória alocada da rotina R_D otimizada por decomposição LU em escala logarítmica

Fonte: Autor

Na Tabela 12 e na Figura 21, são apresentados os valores de tempo de processamento da rotina RD com a otimização por decomposição LU.

Tabela 12 - Tempo de processamento da rotina R_D otimizada por decomposição LU

Fonte: Autor

Circuitos π	Não Otimizado (s)	Otimização SPS (s)	Otimização LU (s)
50	0,061	0,076	0,157
100	0,089	0,084	0,404
200	0,168	0,139	1,342
400	1,934	0,643	6,858
600	4,495	2,660	14,821
800	8,209	5,174	24,919
1000	12,561	8,605	39,776
1200	18,819	12,829	55,999
1400	26,619	18,217	76,002
1600	35,424	25,125	101,011
1800	46,760	33,388	130,995
2000	59,414	43,256	163,366

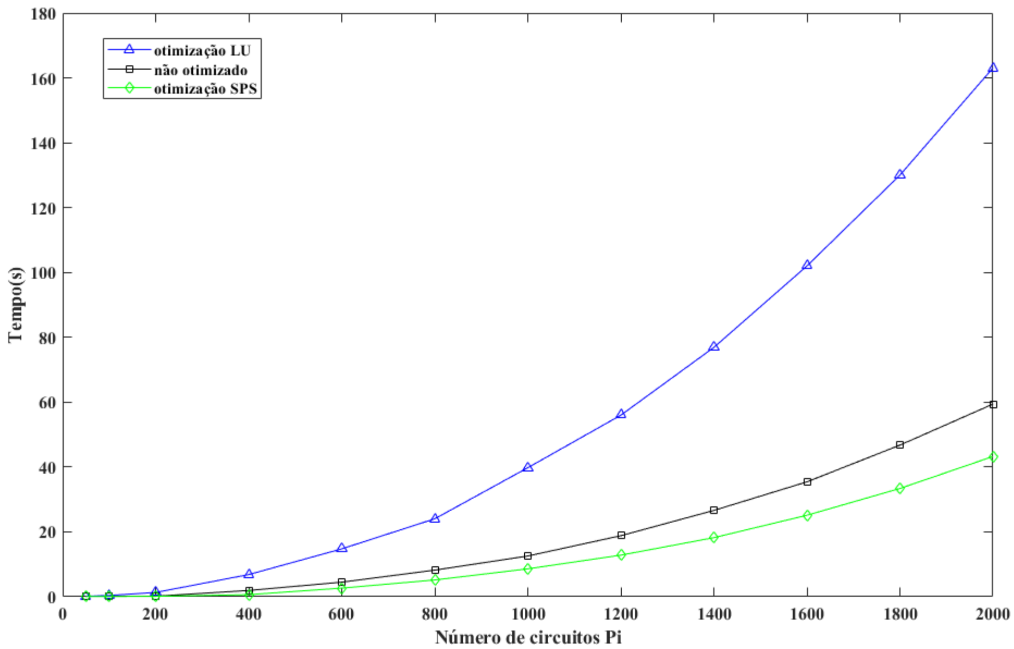

Figura 21 - Gráfico do tempo de processamento da rotina R_D otimizada por decomposição LU

Fonte: Autor

A partir dos resultados da Tabela 12 e da Figura 21, é possível visualizar que os valores de tempo de processamento da rotina base são muito altos comparados com os resultados da rotina sem otimização e também com a otimização por esparsidade. Os valores de tempos foram tão altos em comparação aos anteriores, pois, a maior parte dos termos dentro da matriz eram nulos e como a técnica faz os cálculos termo a termo, os nulos apenas geravam gasto de tempo.

3.3 OTIMIZAÇÃO POR COMBINAÇÃO DE TÉCNICAS

Nos tópicos anteriores foram utilizados a esparsidade e a decomposição como técnicas de otimização para as duas rotinas do trabalho. A esparsidade apresentou ótimos resultados em relação a memória e tempo de processamento. Já a decomposição LU não se mostrou nem um pouco efetiva com relação a otimização das rotinas, porém, como foi explicado anteriormente, ela não teve sucesso pois a maior parte dos termos dentro da matriz eram nulos e como a técnica faz os cálculos termo a termo, os nulos apenas geravam gasto de tempo, visto que apenas as poucas diagonais da matriz eram importantes para os cálculos.

Neste tópico, será abordada uma combinação entre técnicas, onde utilizaremos tanto a esparsidade quanto a decomposição LU. Para utilizar a decomposição LU nas rotinas, todas a modificações apresentadas no tópico 3.3 serão novamente feitas, porém agora, com a implementação de esparsidade. A matriz A será substituída pela a matriz esparsa da mesma. Em (31), é apresentada a equação não otimizada.

$$A' = inv(Ident - \left(\left(\frac{dt}{2}\right) * A\right)) \tag{31}$$

Após a inserção das duas técnicas, decomposição LU e matriz esparsa de A, é gerada a equação vista em (32).

$$[LL, UU] = lu(SPSIdent - \left(\frac{dt}{2}\right) * SPSA) \tag{32}$$

Com isso, a equação final das rotinas que era dada por (33).

$$x = A' * (A'' * x + B1 * du) \tag{33}$$

Agora, após a otimização será dada por (34).

$$x = UU/(\frac{LL}{A'' * x + B1 * du}) \tag{34}$$

A principal ideia em combinar as técnicas é conciliar a esparsidade, que permite apenas que o valores não nulos da matriz sejam armazenados com a decomposição LU, que agora não tendo valores nulos na matriz, pode acelerar o processo de cálculo. Na Tabela 13 e na Figura 22, são apresentados os valores de memória alocada da rotina base com a otimização por combinação de técnicas comparados aos resultados das técnicas anteriores.

Utilizando a Tabela 13 e a Figura 22, percebe-se que a diminuição de memória alocada da rotina base a partir da combinação de técnicas é a mesma que aquela obtida utilizando apenas a esparsidade. Para valores baixos de circuitos π, a diminuição é em torno de 90%. Porém o resultado significativo é para valores altos de circuitos π, como por exemplo, o valor de 2000 circuitos, onde há diminuição de mais de 99,8%. A técnica de decomposição LU não afeta em nada com relação a memória alocada, fazendo com que, para os resultados de memória alocada, a otimização por combinação de técnicas tem o mesmo resultado da otimização por esparsidade.

Tabela 13 - Memória alocada da rotina base otimizada por combinação de técnicas

Fonte: Autor

Circuitos π	Não Otimizado (Mb)	Otimização SPS (Mb)	Otimização LU (Mb)	Otimização SPS-LU (Mb)
50	0,080	0,005	0,080	0,005
100	0,320	0,011	0,320	0,011
200	1,280	0,022	1,280	0,022
400	5,120	0,045	5,120	0,045
600	11,520	0,067	11,520	0,067
800	20,480	0,090	20,480	0,090
1000	32,000	0,110	32,000	0,110
1200	46,080	0,130	46,080	0,130
1400	62,720	0,160	62,720	0,160
1600	81,920	0,190	81,920	0,190
1800	103,680	0,210	103,680	0,210
2000	128,000	0,230	128,000	0,230

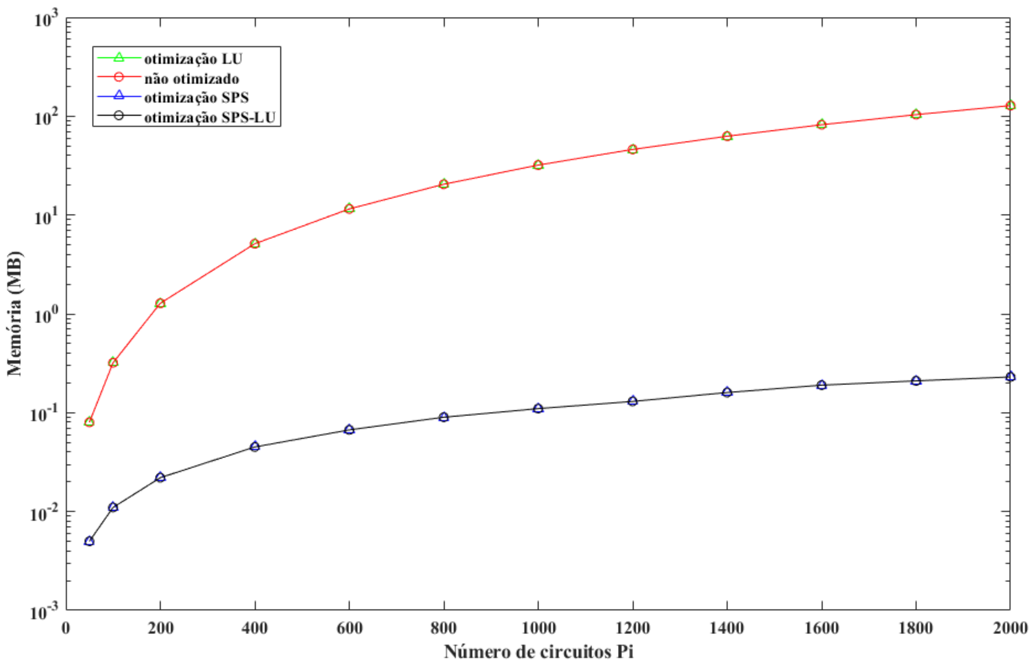

Figura 22 - Gráfico de memória alocada da rotina base otimizada por combinação de técnicas em escala logarítmica

Fonte: Autor

Na Tabela 14 e na Figura 23, são apresentados os valores de tempo de processamento, dados em segundos, da rotina base com a otimização por combinação de técnicas, comparada aos resultados de técnicas anteriores.

Tabela 14 - Tempo de processamento da rotina base otimizada por combinação de técnicas

Fonte: Autor

Circuitos π	Não Otimizado (s)	Otimização SPS (s)	Otimização LU (s)	Otimização SPS-LU (s)
50	0,103	0,077	0,200	0,113
100	0,125	0,087	0,460	0,182
200	0,257	0,148	1,422	0,337
400	2,195	0,503	6,837	0,677
600	5,353	2,838	14,761	1,708
800	9,559	5,186	26,036	2,468
1000	13,946	8,389	40,786	3,233
1200	19,951	11,721	57,989	6,721
1400	28,433	16,396	78,889	8,735
1600	36,924	21,959	102,997	11,275
1800	49,371	29,436	132,071	17,325
2000	61,044	37,851	164,058	21,390

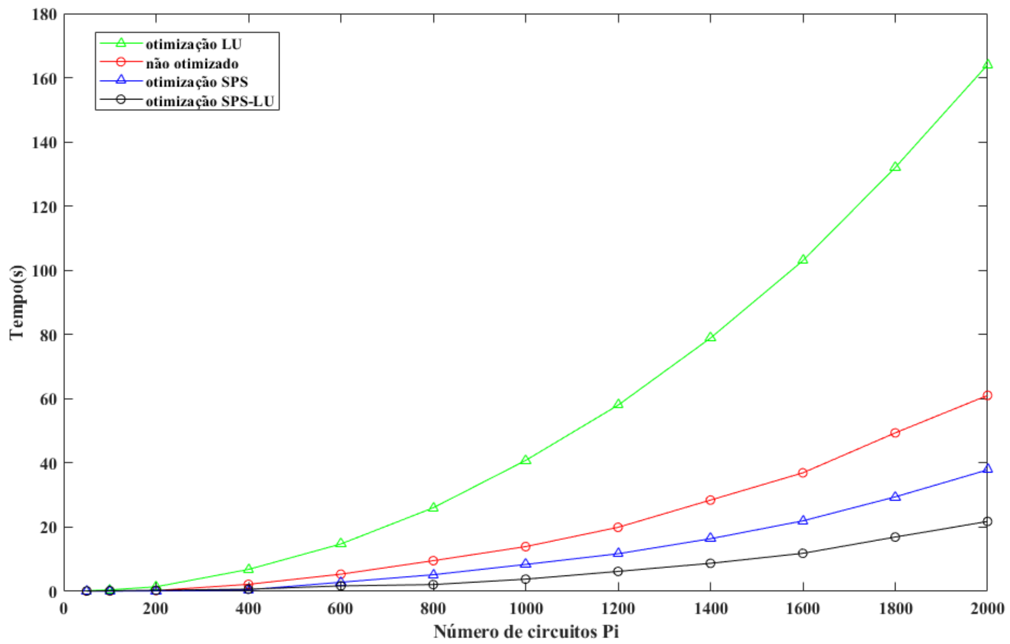

Figura 23 - Gráfico do tempo de processamento da rotina base otimizada por combinação de técnicas

Fonte: Autor

A partir dos valores dados na Tabela 14 e na Figura 23, é possível visualizar que o tempo de processamento da rotina base com a utilização da combinação de técnicas comparado aos resultados apresentados pela rotina não otimizada é insatisfatório até 200 circuitos π. Porém, a partir desta quantidade de circuitos, os resultados passam a ser muito bons chegando a uma diminuição de aproximadamente 65% para 2000 circuitos π.

Já em comparação com a técnica de esparsidade, os resultados da combinação de técnicas são piores até aproximadamente 400 circuitos. A partir disto, a combinação de técnicas se apresenta mais eficaz, chegando a uma diminuição de até 43%. Esta diminuição se dá por causa da decomposição LU que, aliada à esparsidade (que tem o papel de descartar os termos nulos da matriz), torna-se muito mais efetiva para o cálculo de matrizes inversas em matrizes de grande ordem. A comparação desta técnica com a decomposição LU não é tão interessante, pois a decomposição LU se mostrou ineficaz quando inserida sozinha na rotina.

Tabela 15 - Memória alocada da rotina R_D otimizada por combinação de técnicas

Fonte: Autor

Circuitos π	Não Otimizado (Mb)	Otimização SPS (Mb)	Otimização LU (Mb)	Otimização SPS-LU (Mb)
50	0,080	0,007	0,080	0,007
100	0,320	0,014	0,320	0,014
200	1,280	0,028	1,280	0,028
400	5,120	0,057	5,120	0,057
600	11,520	0,086	11,520	0,086
800	20,480	0,115	20,480	0,115
1000	32,000	0,143	32,000	0,143
1200	46,080	0,172	46,080	0,172
1400	62,720	0,201	62,720	0,201
1600	81,920	0,230	81,920	0,230
1800	103,680	0,259	103,680	0,259
2000	128,000	0,287	128,000	0,287

Do mesmo jeito que foi tratada a rotina base, também se faz necessário fazer a apresentação dos dados de alocação de memória e de tempo de processamento para a rotina R_D. Com isso, na Tabela 15 e na Figura 24, são apresentados os valores de memória alocada da rotina R_D.

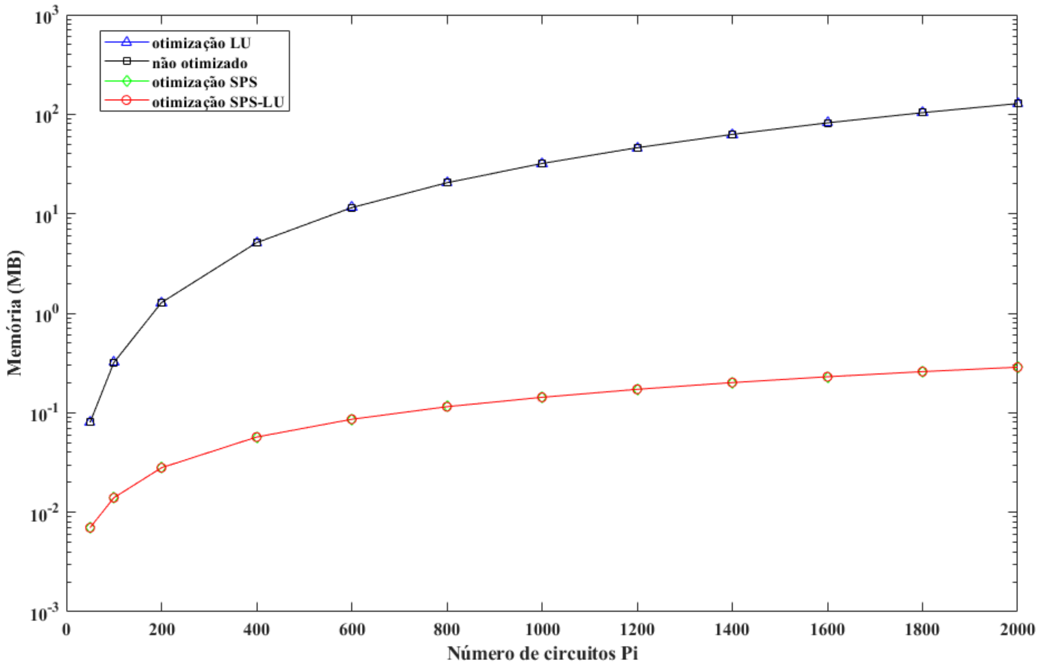

Figura 24 - Gráfico de memória alocada da rotina R_D otimizada por combinação de técnicas em escala logarítmica

Fonte: Autor

A partir da Tabela 15 e da Figura 24, observa-se que a diminuição de memória alocada da rotina R_D a partir da combinação de técnicas é a mesma que aquela obtida utilizando apenas a esparsidade. Para valores baixos de circuitos π a diminuição é em torno de 48%. Porém, o resultado significativo é para valores altos de circuitos π, como por exemplo o valor de 2000 circuitos, onde há uma diminuição de mais de 99,7%. Novamente, a técnica de decomposição LU não afeta em nada com relação a memória alocada, fazendo com que, para os resultados de memória alocada, a otimização por combinação de técnicas tem o mesmo resultado da otimização por esparsidade.

Na Tabela 16 e na Figura 25, são apresentados os valores de tempo de processamento, dados em segundos, da rotina R_D com a otimização por combinação de técnicas comparada aos resultados das técnicas de otimização anteriores e à rotina não otimizada.

Tabela 16 - Tempo de processamento da rotina R_D otimizada por combinação de técnicas

Fonte: Autor

Circuitos π	Não Otimizado (s)	Otimização SPS (s)	Otimização LU (s)	Otimização SPS-LU (s)
50	0,061	0,076	0,157	0,097
100	0,089	0,084	0,404	0,182
200	0,168	0,139	1,342	0,485
400	1,934	0,643	6,858	0,981
600	4,495	2,660	14,821	1,908
800	8,209	5,174	24,919	3,468
1000	12,561	8,605	39,776	5,233
1200	18,819	12,829	55,999	7,721
1400	26,619	18,217	76,002	10,435
1600	35,424	25,125	101,011	14,275
1800	46,760	33,388	130,995	19,325
2000	59,414	43,256	163,366	24,390

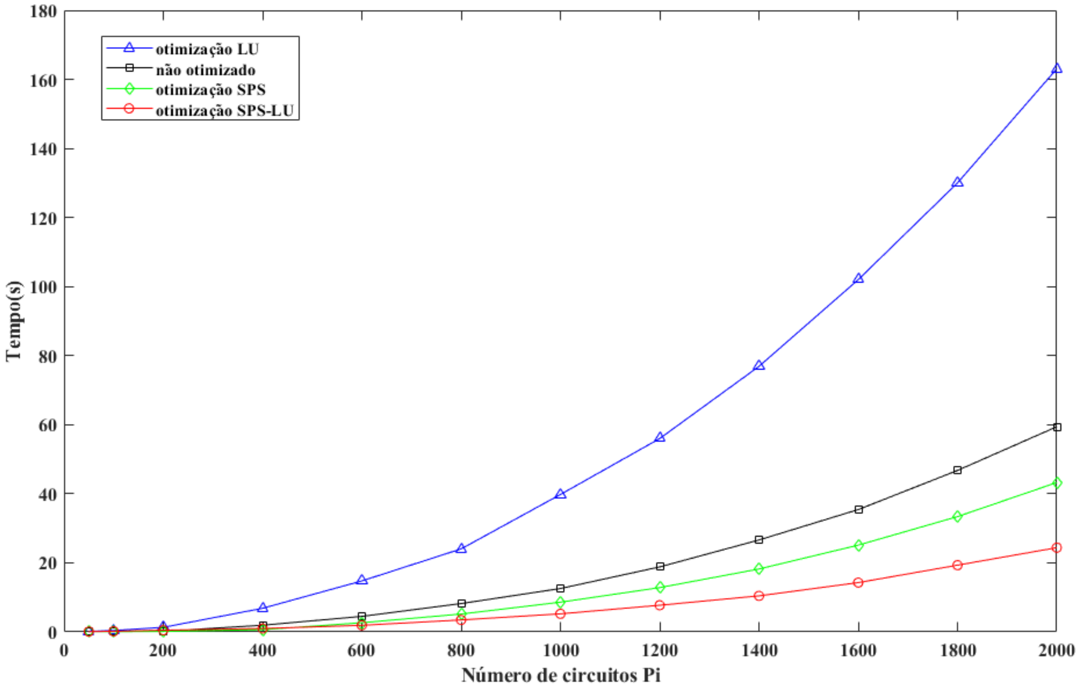

Figura 25 - Gráfico do tempo de processamento da rotina R_D otimizada por combinação de técnicas

Fonte: Autor

A partir dos valores dados na Tabela 16 e na Figura 25, é possível visualizar que o tempo de processamento da rotina R_D com a utilização da combinação de técnicas comparado aos resultados apresentados pela rotina não otimizada é insatisfatório até 200 circuitos π. Porém a partir desta quantidade de circuitos, os resultados passam a ser muito bons, chegando a uma diminuição de aproximadamente 58% para 2000 circuitos π.

Já em comparação a técnica de esparsidade, os resultados da combinação de técnicas são piores até, aproximadamente, 400 circuitos π. Depois desse valor, a combinação de técnicas se apresenta mais eficaz, chegando a uma diminuição de até 43%. Esta diminuição se dá por causa da decomposição LU que, aliada à esparsidade (que tem o papel de descartar os termos nulos da matriz), se torna muito mais efetiva para o cálculo de matrizes inversas em matrizes de grande ordem. A comparação desta técnica com a decomposição LU não é tão interessante, pois a decomposição LU se mostrou ineficaz quando inserida sozinha na rotina.

3.4 COMPARAÇÃO DE OTIMIZAÇÕES ENTRE MODELOS

Com todas as técnicas de otimização propostas no início do trabalho já feitas, conclui-se que a melhor delas em relação a diminuição de memória e tempo de processamento foi a que se utiliza da combinação das técnicas. Neste tópico, é abordada uma comparação entre a rotina base e a rotina R_D. Cada uma delas será apresentada com os resultados de suas melhores otimizações, que no caso de ambas é a combinação de técnicas descrita acima.

A Tabela 17 apresenta a comparação entre as rotinas base e R_D com relação à memória alocada de ambas. Todos os valores apresentados na tabela são baseados em valores obtidos nos tópicos anteriores e determinados pelas mesmas quantidades de circuitos π que vão de 50 a 2000.

Os valores dados na Tabela 17 são valores relativamente próximos, dado que, a diferença entre as duas rotinas é a utilização de uma resistência de amortecimento. Aprofundando a análise, a Figura 26 apresenta, graficamente, todos os valores dados pela Tabela 17.

Tabela 17 - Comparação entre rotinas de memória alocada com otimização por combinação de técnicas

Fonte: Autor

Circuitos π	Rotina Base (Mb)	Rotina RD (Mb)
50	0,005	0,007
100	0,011	0,014
200	0,022	0,028
400	0,045	0,057
600	0,067	0,086
800	0,090	0,115
1000	0,110	0,143
1200	0,130	0,172
1400	0,160	0,201
1600	0,190	0,230
1800	0,210	0,259
2000	0,230	0,287

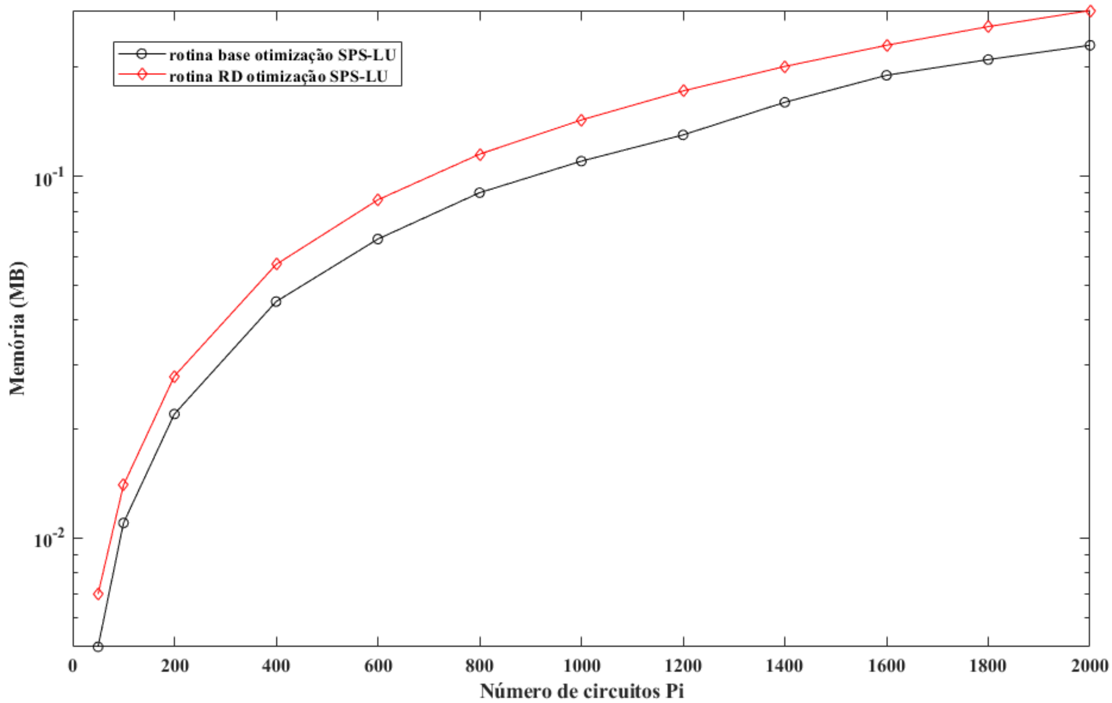

Figura 26 - Gráfico comparativo de memória alocada das rotinas com otimização por combinação de técnicas em escala logarítmica

Fonte: Autor

A partir dos resultados apresentados tanto pela Tabela 17 quanto pela Figura 26, em questão de memória alocada, as duas rotinas estudadas por este trabalho apresentam valores muito próximos. A diferença entre elas é de menos de 0,06 Mb para a maior quantidade de circuitos π apresentada na tabela. Esta diferença tão pequena é dada porque a rotina R_D é uma derivação da própria rotina base, onde, na rotina R_D só é acrescida uma resistência de amortecimento. Com relação as matrizes A internas das duas rotinas, a diferença é de duas diagonais a mais para a rotina R_D, que, ao invés de ter apenas três diagonais com termos não nulos relevantes para os cálculos, terá cinco diagonais de termos não nulos.

Para continuar com a comparação entre as duas rotinas, na Tabela 18 e na Figura 27, são apresentados os valores de tempo de processamento, dados em segundos, da rotina R_D com a otimização por combinação de técnicas, comparada à rotina base com otimização por combinação de técnicas também.

Tabela 18 - Comparação entre rotinas de tempo de processamento com otimização por combinação de técnicas

Fonte: Autor

Circuitos π	Rotina base (s)	Rotina RD (s)
50	0,113	0,097
100	0,182	0,182
200	0,337	0,485
400	0,677	0,981
600	1,708	1,908
800	2,468	3,468
1000	3,233	5,233
1200	6,721	7,721
1400	8,735	10,435
1600	11,275	14,275
1800	17,325	19,325
2000	21,390	24,390

Semelhante ao que foi apresentado na comparação de memória alocada, os resultados de tempo de processamento entre as duas rotinas se mostram muito próximos. Porém, como visto na Tabela 18 e na Figura 27, a diferença entre os resultados aumenta, fazendo com que a rotina base seja mais rápida que a rotina R_D com o aumento de circuitos π. Isso ocorre porque como a rotina R_D possui duas diagonais de termos não nulos a mais que a rotina base e, com o aumento de circuitos π, os termos calculados pela rotina R_D passam a aumentar muito mais do que os calculados pela rotina base. Essa diferença pode ser muito alta para valores altos de circuitos π, mas para os valores analisados neste trabalho, a diferença entre eles chega a no máximo três segundos, não sendo muito significante.

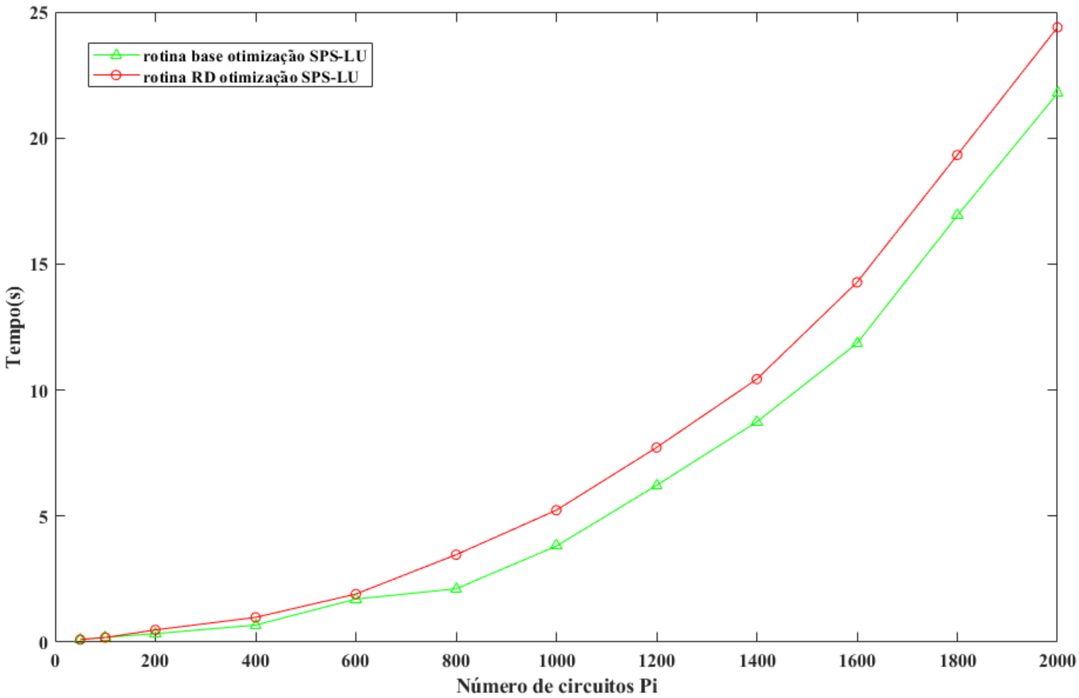

Figura 27 - Gráfico comparativo de tempo de processamento das rotinas com otimização por combinação de técnicas

Fonte: Autor

4. CONCLUSÃO

Em conjunto com o estudo de linhas de transmissão, o estudo de transitórios eletromagnéticos é de grande importância, pois é possível, baseado em simulações digitais prever a propagação de distúrbios que sobrecarregam um sistema. Com isso, sistemas de proteção e coordenação são dimensionados. Com o trabalho proposto, foi possível fazer testes para comparações qualitativas entre otimizações utilizadas em rotinas de simulações de transitórios em linhas de transmissão. Tais otimizações estão relacionadas à diminuição de alocação de memória necessária para modelar a matriz utilizada na representação da linha de transmissão mencionada. Essa diminuição da memória alocada está diretamente relacionada ao tempo de processamento necessário para a realização das simulações desejadas. Assim, a diminuição da memória alocada leva também a um menor tempo de processamento para realização das simulações mencionadas.

O método que se mostrou mais eficaz em ambas as rotinas utilizadas neste trabalho foi o de otimização utilizando a decomposição LU juntamente com esparsidade. Ele se faz muito eficaz com relação a tempo de processamento, apresentando diminuições de quase 43% do tempo para simulações com 2000 circuitos π em comparação as rotinas que se utilizam apenas da técnica de esparsidade como otimização. Em comparação as rotinas sem nenhum tipo de otimização, ela apresenta uma diminuição de aproximadamente 60% para simulações com 2000 ou mais circuitos π.

Com relação à memória reservada, este método não trouxe nenhuma melhoria em relação as rotinas que se utilizam apenas da técnica de esparsidade, pois apenas a técnica de esparsidade influencia a alocação memória já que é ela a responsável na diminuição dos termos que serão reservados. Em comparação as rotinas sem nenhum tipo de otimização, ela apresenta uma diminuição de mais de 99,8% de memória alocada para simulações com 2000 circuitos π na rotina base e, na rotina R_D, uma diminuição de aproximadamente 99,7%. Um ponto importante é que só é interessante a utilização da decomposição LU para as simulações em conjunto com a esparsidade, pois ao se utilizar esta técnica sem a esparsidade, ela faz cálculos com todos os termos nulos inclusos, fazendo com que este processo seja mais demorado do que calcular a matriz inversa diretamente.

REFERÊNCIAS BIBLIOGRÁFICAS

(1) Dommel, H. W.; Yan, A.; Ortiz de Marcano, R. J.; Miliani, A. B. (1983). **"Case Studies for Electromagnetic Transients"**, University of British Columbia, Vancouver, Canada.

(2) Dommel, H. W. (1969). **"Digital computer solution of electromagnetic transients in single and multiphase networks"**, IEEE Trans. on Power Apparatus and Systems, vol. PAS-88, pp. 388-399.

(3) Marti, J. R. (1982). **"Accurate modelem of frequency-dependent transmission lines in 105 electromagnetic transient simulations"**, IEEE Trans. on Power Apparatus and Systems, vol. PAS-101, nº 1, pp. 147-155.

(4) Dommel, H.W. (1986). **"*Electromagnetic Transients Program. Reference Manual (EMTP Theory Book*)"**, Bonneville Power Administration, Portland.

(5) Microtran, Reference Manual (1992). **"Transients Analysis Program for Power and Power Electronic Circuits"**, Microtran Power System Analysis Corporation, Vancouver, B.C., Canada.

(6) *ATPDRAW version 3*, User Manual (1996). TR A4389, EFI, Norway.

(7) Nelms, R. M.; Sheble, G. B.; Newton, S. R.; Grigsby, L. L. (1989). **"Using a personal computer to teach power system transients"**, IEEE Transactions on Power Systems, vol. 4, nº 3, 1293-1297.

(8) Macías, J. A. R.; Expósito, A. G.; Soler, A. B. (2005). **"A Comparison of Techniques for Statespace Transient Analysis of Transmission Lines"**, IEEE Transactions on Power Delivery, vol. 20, nº 2, pp. 894-903.

(9) Mamis, M. S.; Nacaroglu, A.; (2003). **"Transient Voltage and Current Distributions on Transmission Lines",** IEE Proceedings Generation, Transmission and Distribution, vol. 149, N0. 6; pp. 705-712.

(10) Mamis, M. S. (2003). **"Computing of Electromagnetic Transients on Transmission Lines with Nonlinear Components",** IEE Proceedings Generation, Transmission and Distribution, vol, 150, N0. 2; PP. 200-203.

(11) Ingelbinck, T.; Andrade, P. R.; Lessa, L. S.; Prado, A. J.; Bovolato, L. F.; Pissolato Filho, J. (2013). **"Analyses of the frequency-dependent model's inductances' resistances for transmission lines"**, CLAGTEE 2013, CD-ROM, 5 pp, Viña del Mar, Chile.

(12) Andrade, P. R.; Monzani, R. C.; Lessa, L. S.; Ingelbinck, T.; Prado, A. J.; Pissolato Filho, J.; Bovolato, L. F. (2013). **"Physical model for representing transmission lines by undergraduate students"**, CLAGTEE 2013, CD-ROM, 5 pp, Viña del Mar, Chile.

(13) Dommel, H. W. (1997). **"Techniques For Analyzing Electromagnetic Transients"**, IEEE Computer Applications on Power, vol. 10, no, 3, pp.

(14) Nguyen, H. V.; Dommel, H. W.; Martí, J. R. (1997). **"Modelling of single-phase non uniform transmission lines in electromagnetic transient simulations"**, IEEE Transactions on Power Delivery, vol.12, no. 2, pp. 916-921.

(15) Justino, L. **"Método da Decomposição LU para a solução de equações lineares"** Disponível em: <http://www.facom.ufu.br/~dino/disciplinas/GBC051/Decomp_LU_Lucas.pdf>. Acesso em: 04 de setembro de 2020.

(16) Roman, N. T.; Digiampietri, L. A. **"Estrutura de dados: Matriz esparsa"** Disponível em: <http://www.each.usp.br/digiampietri/ed/aula14.pdf>. Acesso em: 04 de setembro de 2020.

(17) **Sparse Matrix Conversion**. SciLab. Disponível em: <https://help.scilab.org/docs/5.4.1/pt_BR/sparse.html>. Acesso em: 04 de setembro de 2020.