**CENTRO FEDERAL DE EDUCAÇÃO TECNOLÓGICA CELSO SUCKOW DA FONSECA**

**DEPARTAMENTO DE ENGENHARIA MECÂNICA**

**BRUNO MELLO SILVEIRA**

**DESENVOLVIMENTO DE CÓDIGO NUMÉRICO PARA ANÁLISE DE VISCOELASTICIDADE EM TECIDOS MOLES**

**TRABALHO DE CONCLUSÃO DE CURSO**

**RIO DE JANEIRO**

**2021**

**BRUNO MELLO SILVEIRA**

**DESENVOLVIMENTO DE CÓDIGO NUMÉRICO PARA ANÁLISE DE VISCOELASTICIDADE EM TECIDOS MOLES**

Trabalho de Conclusão de Curso apresentado como requisito parcial à obtenção do título de Bacharel em Engenharia Mecânica, do Departamento de Engenharia Mecânica, do Centro Federal de Educação Tecnológica Celso Suckow da Fonseca.

Orientador: Prof. Dr. Paulo Pedro Kenedi

**RIO DE JANEIRO**

**2021**

Resumo

SILVEIRA, Bruno Mello. **Desenvolvimento de código numérico para análise de viscoelasticidade em tecidos moles.** 2021. Número total de folhas. Trabalho de Conclusão de Curso – Centro Federal de Educação Tecnológica Celso Suckow da Fonseca. Rio de Janeiro. Rio de Janeiro, 2021.

Para se calcular a evolução das tensões em tecidos moles, faz-se necessário utilizar equações que não possuem soluções analíticas fechadas, sendo, portanto, fundamental o uso de métodos numéricos para conseguir resolvê-las. Este trabalho busca desenvolver um código numérico nos moldes de uma API, sendo capaz de realizar cálculos baseando-se nos modelos de viscoelasticidade linear de Maxwell e quase-linear de Fung, gerando ao fim, um arquivo em formato CSV com os resultados para comparar com os dados experimentais. Esses dados são obtidos a partir de ligamentos de joelho de porco como: ligamento colateral lateral (LCL), ligamento cruzado anterior (LCA), ligamento cruzado posterior (LCP) e ligamento colateral medial (LCM).

**Palavras-chave:** viscoelasticidade, ligamento de joelho, código numérico, API.

abstract

SILVEIRA, Bruno Mello. **Numerical code development for viscoelasticity analysis in soft tissue.** 2021. Número total de folhas. Trabalho de Conclusão de Curso - Federal Center of Technological Education – Rio de Janeiro. Rio de Janeiro, 2021.

To calculate the stress Evolution in soft tissues, it is necessary to use equations that do not have closed analytical solutions, therefore, it is essential to use numerical methods to solve them. This work seeks to develop a numerical code based on API model, being able to perform calculations based on Maxwell's linear and Fung's quasi-linear viscoelastic models, creating, at the end, a file in CSV format with the results to compare with experimental data. These data are captured from porcine knee ligaments such as: lateral collateral ligament (LCL), anterior cruciate ligament (ACL), posterior cruciate ligament (PCL) and medial collateral ligament (MCL).

**Keywords:** viscoelasticity, knee Ligament, numerical code, API.

LISTA DE ILUSTRAÇÕES

Figura 1 - Representação de um joelho humano (WISMANS, 1980).........................7

Figura 2 - Tensão e deformação em uma situação de (a) fluência e de (b) relaxação..10

Figura 3 – Exemplo de texto na formatação JSON.....................................................14

Figura 4 – Formas de passar o conjunto de dados da requisição no *Swagger*..........15

Figura 5 – *Swagger* da API desenvolvida para este estudo.......................................15

Figura 6 – Exemplo de código presente no *controller*.................................................17

Figura 7 – Fluxograma básico das operações.............................................................18

Figura 8 – Representação de mola e amortecedor em série......................................23

Figura 9 – Força e alongamento ao longo do tempo para (a) relaxação e para (b) fluência.......................................................................................................................26

Figura 10 – Modelo generalizado de Maxwell com nomenclatura adaptada................27

Figura 11 - Gráfico de deformação por tempo............................................................29

Figura 12 – Fluxograma das operações de (a) calcular os resultados do modelo e (b) fazer análise de sensibilidade das variáveis...............................................................37

Figura 13 – Fluxograma para as sub-rotinas que calculam a tensão e deformação....38

Figura 14 – Fluxograma da sub-rotina que calcula os resultados para modelo Quase-Linear..........................................................................................................................40

sumário

[1 introdução 6](#_Toc73355152)

[1.1 Revisão bibliográfica 6](#_Toc73355153)

[1.2 conceitos fundamentais de viscoelasticidade 10](#_Toc73355154)

[1.3 Conceitos fundamentais de programação 12](#_Toc73355155)

[2 RECURSOS implementados 20](#_Toc73355156)

[2.1 métodos numéricos 20](#_Toc73355157)

[2.2 dados experimentais 22](#_Toc73355158)

[2.3 operações auxiliares 22](#_Toc73355159)

[3 modelo linear de viscoleasticidade 23](#_Toc73355160)

[3.1 maxwell 23](#_Toc73355161)

[3.2 maxwell generalizado 26](#_Toc73355162)

[4 modelo quase-linear de viscoelasticidade 28](#_Toc73355163)

[4.1 Equacionamento 28](#_Toc73355164)

[4.1.1 Deformação 29](#_Toc73355165)

[4.1.2 Resposta elástica (parcela elástica) 30](#_Toc73355166)

[4.1.3 Função relaxação reduzida (parcela viscosa) 31](#_Toc73355167)

[4.1.4 Tensão 34](#_Toc73355168)

[5 implementação numérica 36](#_Toc73355169)

[5.1 modelo linear 38](#_Toc73355170)

[5.2 modelo quase-linear 39](#_Toc73355171)

[REFERÊNCIAS 41](#_Toc73355172)

[anexo A - Descrição das equações usadas para modelo de viscoelasticidade Quase-Linear 46](#_Toc73355173)

1. introdução

O presente trabalho busca desenvolver um código numérico utilizando linguagem C# nos moldes de uma API capaz de realizar cálculos de viscoelasticidade aplicada a tecidos moles, baseando-se nos modelos linear de Maxwell, quase-linear de Fung e não-linear de Schapery, além de gerar um arquivo externo em formato CSV, para viabilizar a comparação dos resultados numéricos com os resultados experimentais. Busca-se com este projeto auxiliar a pesquisa sobre viscoelasticidade em tecidos moles desenvolvida pelo professor Paulo Pedro Kenedi, sendo apresentada os cálculos, as rotinas e as operações secundárias necessárias.

## Revisão bibliográfica

No ramo da indústria civil, pode-se ver a aplicação da viscoelasticidade em modelos de fluência de estruturas mistas de aço, conforme apresentado por (TARECO, 2014) que usa os modelos lineares de Maxwell e de Kelvin para modelar uma estrutura de aço-betão, analisando a relaxação e a fluência somente do betão na resposta da estrutura mista e dos materiais que a compõem. Além disso, conforme apresentado (QUEIROZ, 2008) materiais viscoelásticos também são utilizados para atenuar vibrações e ruídos em estruturas, tendo aplicações tanto no setor automotivo quanto aeroespacial. Com o objetivo de melhor compreender os modelos, (R. F. Navarro, 2017) aborda as combinações possíveis entre as analogias mecânicas mola e amortecedor que caracterizam o comportamento viscoelásticos de materiais reais de uso no ramo industrial.

Pode ver também a viscoelasticidade aplicada a outros materiais, como visto na obra de (WEINEROWSKA-BORDS, 2015) que analisou os aspectos relativos ao comportamento viscoelástico de paredes de tubos de polímero durante um fluxo instável, também apresentou, para o modelo linear, a expressão que utiliza convolução do termo viscoelástico e comparou esta com a equação correspondente ao atrito instável.

A importância do estudo da viscoelasticidade não se restringe apenas à metais e polímeros, tendo também importâncias no estudo de tendões e ligamentos, conhecidos como tecidos moles, como mostrado por (ROSSETTO, 2009) que reforça que esse conhecimento é de suma importância para melhor compreensão do comportamento dos tendões, permitindo que melhores análises possam ser feitas para determinar o treinamento físico, como nos casos de terapêuticos, das tendinopatias. (BERNARDES, et. al, 2005), buscou determinar os parâmetros biomecânicos para o modelamento da articulação do joelho humano, conforme mostrado esquematicamente na Fig.1, através de exercícios extensivos, junto de imagens obtidas por videofluoroscópio, onde a viscoelasticidade desempenha um importante papel. (ZHENG, et al., 1998) buscou modelar as forças internas em um joelho humano, analiticamente, durante um exercício físico, além disso, usaram um software para otimizar os resultados e minimizar os erros.

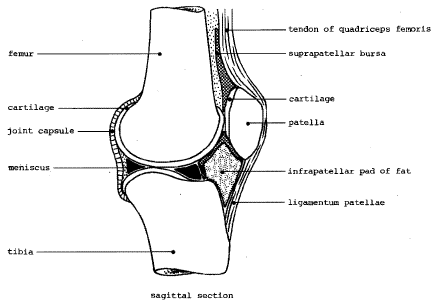
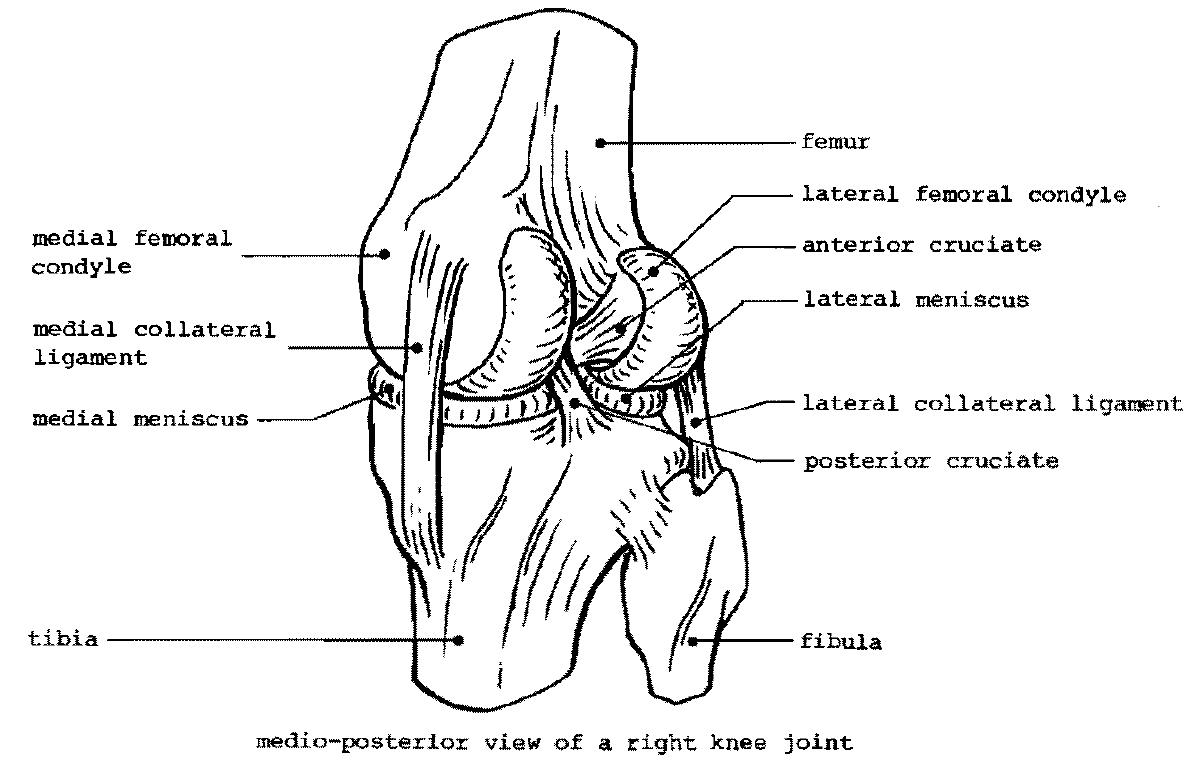


Figura 1 – Representação de um joelho humano (WISMANS, 1980)

Devido a sua simplicidade, algumas pesquisas foram feitas com o objetivo de caracterizar as propriedades mecânicas de tecidos moles por meio de modelos lineares de viscoelasticidade. Isto foi feito em (JAMISON, et al., 1968) em que é apresentado um estudo caracterizando os tecidos moles via viscoelasticidade linear, apesar do modelo teórico não ser tão compatível com o comportamento real, visto pela investigação experimental dos autores com pele de porco. Na obra de (MILLER, 1998), também é usado o modelo linear com o objetivo de aplicar em um software de elementos finitos para caracterizar as propriedades mecânicas de tecidos de cérebro.

Como os modelos lineares são mais simples, mas não refletem o comportamento real de muitos materiais viscoelásticos, alguns autores preferiram utilizar o modelo de viscoelasticidade quase-linear apresentado por (Fung, 1993) ou mesmo modelos não-lineares mais complexos, por se aproximarem mais da realidade. Em (PIAZZA, et al., 2001) desenvolveram um modelo dinâmico tridimensional da articulação tibiofemoral e patelofemoral para prever o movimento de implantes no joelho durante atividades intensas se baseando no modelo de Fung, em que se utilizou as equações dinâmicas de movimento sujeitas às forças geradas pelos músculos, ligamentos e contato nas articulações tibiofemoral e patelofemoral. Além disso, (DEBSKI, et al., 2004) usaram o modelo quase-linear para obter uma função de relaxamento que melhor se adequasse aos tecidos moles.

(XU e ENGQUIST, 2018) propuseram um modelo matemático para o módulo de relaxação e sua solução numérica, se baseando em modelos não-lineares de viscoelasticidade. Este modelo apresentou-se capaz de simular fluência, deformação sinusoidal e dissipação de energia, além de apresentar melhor precisão no ajuste de dados experimentais e estabilidade de predição fora da faixa experimental quando comparado a outros modelos amplamente utilizados, como a série de Prony, presente no modelo de Maxwell generalizado, que será detalhada em próximos capítulos deste trabalho.

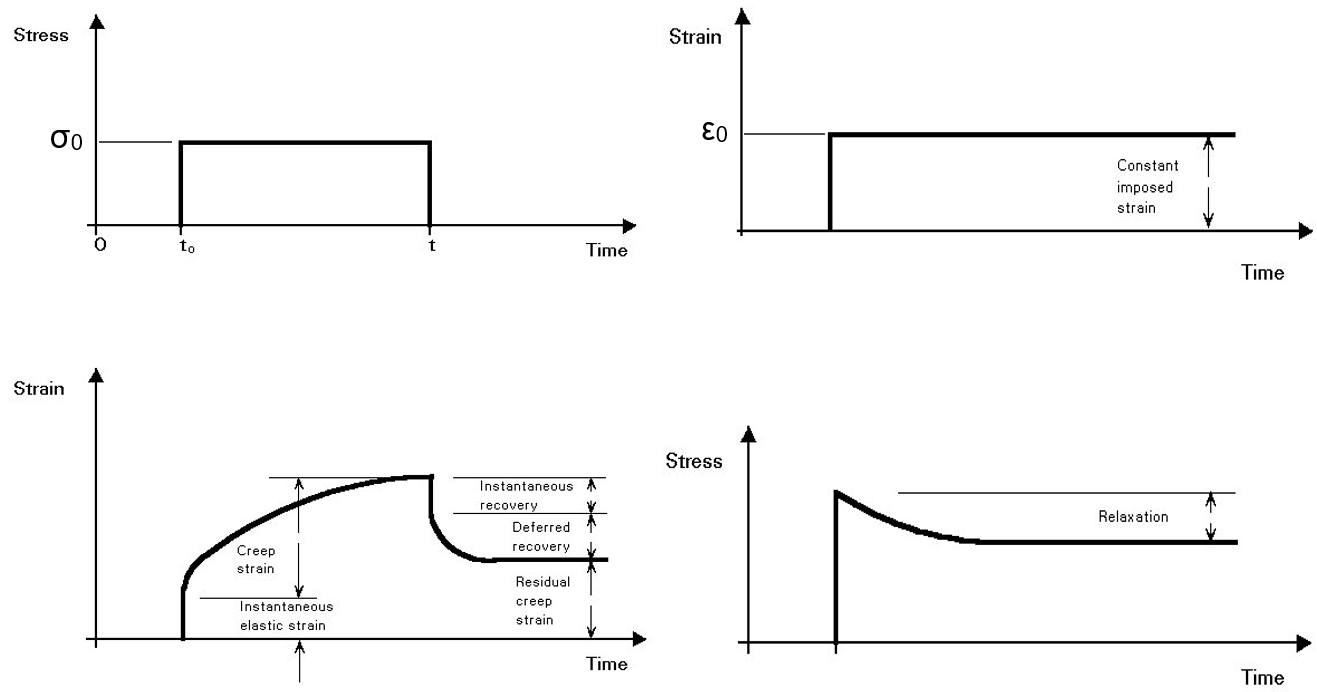
Com o objetivo de analisar o comportamento viscoelástico de tendões ao longo do tempo, diversos autores fizeram variados experimentos de modo a observar o comportamento tanto da tensão e quanto da deformação. (BONIFASI-LISTA, et al., 2004) quantificaram a deformação do ligamento medial colateral (MCL) de um ser humano aplicando tensões nas direções longitudinais e transversais, e sobre cisalhamento ao longo da direção das fibras. Também verificaram se o comportamento viscoelástico depende da direção de aplicação. (DUENWALD, et al., 2009) utilizaram o ligamento medial colateral (MCL) impondo deformações de 1, 2, 3, 4, 5 e 6% com o objetivo de explorar a dependência com o tempo da recuperação deste ligamento, já que, de acordo com estes, é uma característica muito importante, porém pouco estudada. Além disso, para que a o efeito da recuperação fosse observado, utilizaram vários patamares de deformações iniciais. Eles observaram que o relaxamento encontrado contrasta com o modelo de viscoelasticidade quase-linear (QLV), que prevê dependência com o tempo igual para várias deformações iniciais impostas, e que os ligamentos não se recuperaram até os níveis previstos pelos modelos não linear e quase-linear, embora tenham tido recuperação parcial.

A preocupação em utilizar recursos computacionais para modelos de viscoelasticidade pode ser visto em diversos artigos, já que se faz necessário executar cálculos complexos que, em alguns casos, não possuem opções para serem executadas analiticamente. Em (DE BARROS, 2020), é abordado os conceitos primordiais para implementar matematicamente a abordagem viscoelástica em tecidos moles para descrever seu comportamento, além de uma revisão bibliográfica detalhada para atingir esse objetivo. (WEISS, et al, 2001), fizeram uma revisão sobre técnicas antigas e atuais para modelar computacionalmente ligamentos e tendões, mostrando conceitos relevantes sobre mecânica do contínuo e elementos finitos, além de dar ênfase na influência microestrutural dos tecidos moles. (ABRAMOWITCH, et al., 2004) buscaram obter as cinco constantes para o modelo teórico quase-linear de viscoelasticidade que são usados para descrever a resposta elástica (constantes *A* e *B*) e a função relaxação reduzida (constantes *C*,  e ) através de uma abordagem que converge para uma única solução com mínima variação. Submeteram seis ligamentos LCL de cabra a testes de tensão uniaxial com tempo de rampa na faixa de 18 segundos seguido de uma hora de relaxação, por fim, usaram as equações constitutivas do modelo QLV com convolução para os ajustes de curva. Nos testes realizados, a convergência falhou para três dos seis ligamentos, com os maiores erros nas constantes *A*, *B* e . (DUENWALD, et al, 2009) utilizaram métodos numéricos para expandir a janela experimental de tempo para o relaxamento e fluência e mostraram as vantagens associadas a isso, também apresentaram um estudado sobre o papel do tempo de rampa, relaxação e recuperação no contexto das formulações para os modelos de viscoelasticidade quase-linear e não-linear. (BABAEI, et al, 2015) desenvolveram um algoritmo eficiente que permite a identificação de espectros de relaxação viscoelástica, sendo testado contra dados experimentais para caracterizar sua robustez e identificar suas limitações e pontos fortes, dessa forma, aplicaram para identificar a resposta viscoelástica do colágeno reconstituído. (DE PASCALIS, et al., 2014) reavaliaram o modelo de Fung para viscoelasticidade quase-linear expondo uma abordagem diferente que produz um comportamento melhorado e oferece um esquema simples para resolver uma ampla gama de modelos por meio de soluções numéricas para a equação de Volterra, além disso, apresentaram que uma série de características negativas encontradas em outras obras, que são atribuídas à Fung, são somente uma consequência da forma como o modelo é aplicado.

## conceitos fundamentais de viscoelasticidade

Entende-se como viscoelasticidade a propriedade dos materiais que apresentam comportamento viscoso e elástico ao mesmo tempo, sendo um conceito amplamente utilizado em diversos setores da indústria. Elasticidade, por exemplo, é a propriedade de um corpo de sofrer deformação quando submetido a carga e retornar ao seu estado de equilíbrio inicial ao cessar a carga. Por outro lado, viscosidade é a resistência de um fluido quanto ao escoamento, causando uma velocidade de fluxo maior quanto mais afastado da superfície de contato. Atualmente há diversos modelos que descrevem o comportamento viscoelástico de diferentes materiais, podendo serem divididos, de forma didática, em linear, quase-linear e não-linear.

Alguns conceitos de viscoelasticidade são abordados, como a fluência e a relaxação. A fluência (*creep*, em inglês) ocorre quando aplicada uma tensão constante ao corpo (por exemplo através da aplicação de um peso morto) e observa-se uma contínua deformação ao longo do tempo, conforme pode ser visto na Fig. 2.a. Por outro lado, a relaxação (*relaxation*, em inglês) ocorre quando aplicada uma deformação constante ao corpo e observa-se a tensão diminuir ao longo do tempo até um determinado patamar e se mantendo constante depois, conforme mostrado na Fig 2.b.



(a) (b)

Figura 2 – (a) Gráficos de tensão x tempo e de deformação x tempo em uma situação de: (a) fluência e (b) relaxação (COMPOSITE CONSTRUCTION, 2020)

O comportamento linear da deformação no caso de fluência pode ser expresso de acordo com a equação 1, que indica que a tensão é constante e a função fluência, expressa por J(t), depende somente do tempo.

|  |  |
| --- | --- |
|  | (1) |

Já para a tensão, seu comportamento linear no caso de relaxação pode ser expresso de acordo com a equação 2, mostrando que a deformação atua de maneira constante e a função relaxação, expressa por G(t), depende somente do tempo.

|  |  |
| --- | --- |
|  | (2) |

Outro conceito importante é o princípio da superposição de Boltzmann que é utilizado em diversos modelos de viscoelasticidade. Segundo este princípio, para o caso unidimensional é considerado uma força F(t) e um alongamento u(t), que é causada por todo o histórico do carregamento até o tempo *t*. Se a função F(t) for contínua e diferenciável, então, em um pequeno intervalo de tempo dτ no tempo *t*, o incremento do carregamento pode ser expresso por (dF/dτ)dτ. Este incremento continua atuando no material e contribui um elemento du(t) para o alongamento no tempo *t*, com uma constante de proporcionalidade *c*, também conhecida como função fluência, dependente do intervalo de tempo t – τ. Isto está descrito na equação 3.

|  |  |
| --- | --- |
|  | (3) |

Algo similar pode ser escrito relacionando o comportamento do carregamento a partir da ação de um alongamento, conforme apresentado na equação 4, contudo, para esta, será usado uma constante de proporcionalidade k, também conhecida como função relaxação.

|  |  |
| --- | --- |
|  | (4) |

Essas formulações também podem ser aplicadas para se obter uma relação entre a deformação e a tensão, já que isto possui uma importância maior para os estudos de viscoelasticidade em tecidos moles, sendo estas descritas a seguir. Contudo, vale lembrar que as constantes de proporcionalidades serão diferentes, sendo estas *J* e *G* que também representam as equações para, respectivamente, a fluência e a relaxação, porém neste caso relacionando deformação e tensão.

|  |  |
| --- | --- |
|  | (5) |
|  | (6) |

De acordo com a hipótese de Boltzmann, para o cálculo do efeito total do carregamento na estrutura analisada pode-se somar cada diferencial, aplicando uma integral do tempo inicial igual a 0 até o tempo t, conforme descrito em seguida.

|  |  |
| --- | --- |
|  | (7) |
|  | (8) |

Essas equações são lineares, já que estas dependem somente do tempo, entretanto, também são vistas utilizadas de maneira adaptada em modelos quase-lineares e não-lineares.

## Conceitos fundamentais de programação

Neste item, diversos conceitos da área de ciência da computação serão explicados. A linguagem de programação escolhida, a linguagem C# que, segundo (Microsoft, 2021), “é uma linguagem de programação moderna, orientada a objeto e de tipo seguro”. Esta linguagem de programação é comumente utilizada no desenvolvimento de sistemas por empresas por seu poder computacional, além de possuir uma vasta documentação ofertada gratuitamente pela *Microsoft*.

O próximo conceito é a API, sigla para *Application Programming* *Interface*, que reúne diversos conceitos. Segundo a Microsoft, a API (Microsoft, 2019) “específica como os componentes e sistemas de software devem interagir uns com os outros”. Segundo (TechTudo, 2020), corresponde a um conjunto de normas que possibilitam a comunicação entre sistemas através de uma série de padrões e protocolos. Um exemplo de cada é o padrão SOLID, que expõe princípios de boas práticas de programação orientada a objeto (POO) de modo a tornar o código limpo, de fácil manutenção e escalável, e o protocolo HTTP, que determina a estrutura básica de comunicação a ser usada por aplicações WEB. Vale salientar que esse padrão e esse protocolo foram utilizados no desenvolvimento do código para este trabalho, com o objetivo de criar um código com grande poder computacional, de fácil compreensão e manutenibilidade.

Por utilizar os protocolos HTTP, pode-se chamar a aplicação desenvolvida para este estudo de API RESTful. De acordo com (HOSTGATOR, 2019 e REDHAT, 2021) A sigla REST significa *Representational State Transfer*, que traduzido para o português significa Transferência Representacional de Estado, e, por sua utilização por completo na API, recebe o sufixo “ful”. Devido a isso, foram utilizados alguns recursos essenciais para o correto funcionamento da API RESTful, como entrada e saída de dados usando formatação JSON, chamadas HTTP através de *endpoint*, que possuem verbos HTTP específicos para cada tipo de tarefa, status HTTP na resposta da operação e estrutura de criação de URLs estáticas, sendo escolhido o *framework* ASP.NET MVC. Como diversos conceitos são utilizados neste contexto, é necessária uma melhor explicação sobre cada um deles.

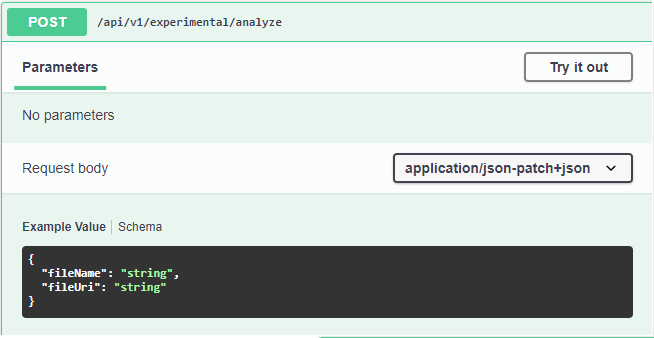
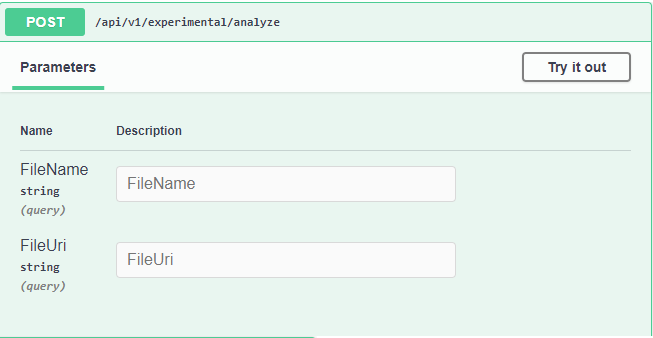
JSON (INTRODUCING, 2021), sigla para *JavaScript Object Notation* ou, em português, Notação de Objetos JavaScript, é uma formatação leve de troca de dados, simples de ler e escrever para o ser humano e de fácil interpretação e geração para máquinas, é independente de linguagem, por usar convenções que são familiares a diversas linguagens de programação, conforme pode ser visto na Fig. 3. No presente trabalho, será amplamente utilizada na serialização de objetos, isto é, converter uma cadeia de texto em um objeto utilizado internamente pelo código, e desserialização de objetos que é o caminho inverso da serialização. Ambos são usados, principalmente, pelo *Swagger* na entrada e saída de dados, respectivamente, conceito que será melhor apresentado mais a diante.

Texto

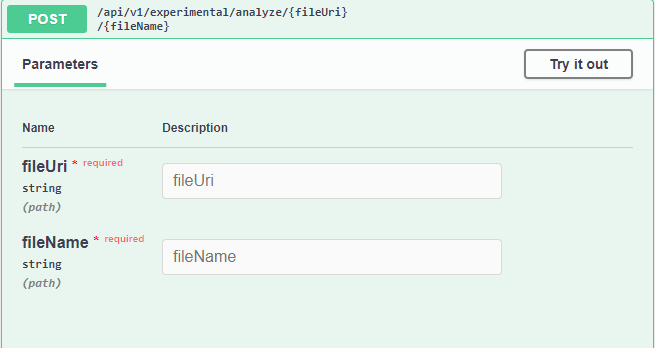
Descrição gerada automaticamente

Figura 3 – Exemplo de texto na formatação JSON

O próximo conceito a ser descrito é o *endpoint* (MILVUS, 2020) e (HOSTGATOR, 2019), esse é um termo em inglês que pode ser traduzido para pontos de extremidade, que, no caso abordado neste documento, possui de fato esta característica, sendo o ponto de extremidade da aplicação. Nesse contexto, o *endpoint* corresponderá a uma URL por onde são feitas as requisições possibilitando que as operações em uma API sejam acessadas. Para efetuar essa requisição, é necessário definir dois parâmetros: o verbo HTTP definido pelo programador, que pode ser GET, para uma busca de dados, POST, para criar um dado, PUT, para atualizar algum dado, e DELETE, para excluir um dado; e o conjunto de dados da requisição passado pelo usuário. Este último pode preenchido de três maneiras diferentes, como pode ser visto na Fig. 4: na URL, escrita juntamente à rota do *enpoint*; por *query*, em que no *Swagger* permite escrever cada parâmetro separado; e no corpo da requisição, usando formatação JSON. Vale salientar que um *endpoint* está atrelado a uma operação que é uma classe que possui as lógicas para a execução de uma determinada rotina e/ou tarefa.

(a) Corpo da requisição no formato JSON (b) Dados de requisição por *query*



(c) Dados de requisição a partir da URL

Figura 4 – Formas de passar o conjunto de dados da requisição no *Swagger*

O próximo conceito é o *Swagger* em que segundo (SMARTBEAR, 2021) é uma ferramenta *open source* para desenvolvimento de APIs, facilitando no design, construção, documentação, teste e padronização, além disso permite ao usuário acessar os *endpoints* disponíveis através de uma interface simples e de fácil compreensão com documentações, exemplos e explicações sobre os recursos, conforme pode ser visto na Fig. 5.

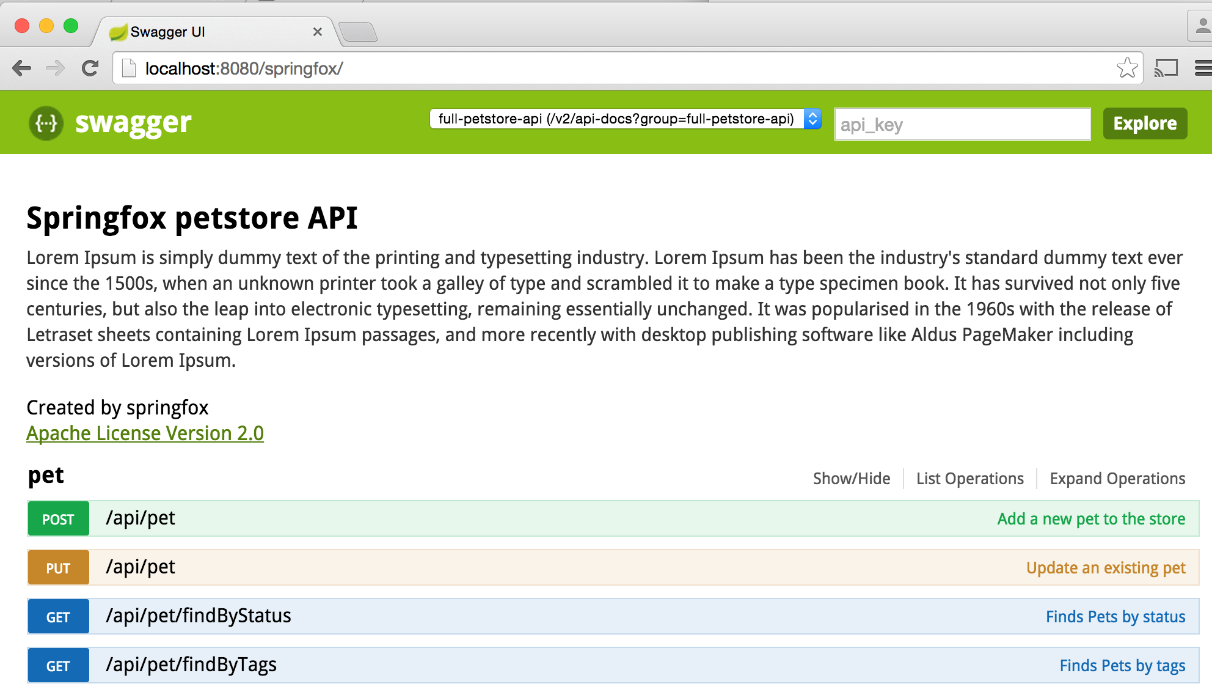


Figura 5 – *Swagger* da API desenvolvida para este estudo (Stacker Overflow, 2018)

Além disso, este recurso permite expor ao usuário a resposta detalhada da operação, que nada mais é do que uma cópia da estrutura da classe cuja responsabilidade única é conter essas informações. No caso deste estudo, todas as classes de resposta possuem a mesma estrutura com as seguintes propriedades: *Success*, que é do tipo booleano, ou seja, pode ser verdadeiro, quando a operação executar corretamente, e falso, caso falhe; *HttpStatusCode*, que contém o status HTTP da operação; *Errors*, que é uma lista onde são escritos os erros que ocorrem durante a aplicação, juntamente a um código pré-definido para facilitar a compreensão do motivo do erro; e *Data*, que contém os dados principais da resposta, variando através da necessidade de cada operação, já que em cada caso há informações diferentes que são importantes de retornar para o usuário.

Devido a sua grande gama de informações, vale informar que somente alguns status HTTP foram utilizados neste estudo, já que este pode informar diversos acontecimentos com a API, conforme mostrado pela empresa Runscope em seu site dedicado a isso. Os valores utilizados são descritos na Tabela 1 com sua nomenclatura, significado e dados de resposta obrigatórios.

Tabela 1 - Status HTTP utilizados no código

|  |  |  |  |
| --- | --- | --- | --- |
| Status HTTP | Nomenclatura | Significado | Dados de resposta obrigatórios |
| 200 | OK | Operação executada com sucesso | Não tem |
| 201 | Created | Arquivo gerado com sucesso | Caminho do arquivo |
| 202 | Accepted | Dados de requisição válidos e aceitos para executar a operação | Não tem |
| 400 | Bad Request | Dados de requisição inválidos | Motivo do erro descrevendo quais dados estão inválidos |
| 401 | Unauthorized | Usuário não tem autorização para acessar o *endpoint* | Não tem |
| 404 | Not found | Nenhum dado foi encontrado de acordo com o filtro passado | Não tem |
| 500 | Internal Server Error | Erro interno | Descrição detalhada do motivo do erro |
| 501 | Not Implemented | Recurso não implementado | Descrição do recurso que não foi implementado |

**Fonte: Runscope, 2021**

Para que a aplicação possa expor na WEB seus recursos, faz-se necessário a utilização de alguma ferramenta que possibilita isso, sendo escolhida para este trabalho o *framework* ASP.NET MVC, que possibilita criar sites estáticos, sendo interpretado com facilidade pelo *Swagger*. Seguindo (Rick Anderson, 2019 e GASPAROTTO, 2014) ASP.NET é uma plataforma da Microsoft criada para desenvolvimento WEB e a sigla MVC vem de *Model View Controller*, sendo este conjunto responsável pela apresentação da aplicação, visando criar um código que não possua conexão forte entre as partes, pois facilita a manutenção e adição de funcionalidades. Dentro deste padrão, há três componentes principais: *model*, responsável por representar a entidade da lógica de negócio da aplicação, ou seja, a operação; *view*, responsável por apresentar uma interface para o usuário, que neste caso estará em conjunto com o *Swagger*; e *controller*, que é responsável pelo controle dos elementos, propiciando uma ligação entre eles. Um exemplo de código escrito no *controller* pode ser visto na Fig. 6, destacando-se em vermelho a parte que representa a operação e em azul a parte que representa os dados de entrada passados pelo usuário através do *Swagger*.

Texto

Descrição gerada automaticamente

Figura 6 – Exemplo de código presente no *controller*

Com o objetivo de simplificar e padronizar o código, foi desenvolvido um conjunto de classes que servem de base para todas as operações, de modo que cada operação da API só precisa implementar seus parâmetros e lógicas específicas, evitando repetições de código. Dessa forma, todas as operações seguem o mesmo fluxograma apresentado na Fig. 7, em que, após a entrada dos dados a partir do *Swagger*, esses são validados no método *ValidateOperationAsync*, caso sejam válidos, segue para o método *ProcessOperationAsync*, onde a operação será de fato executada e, caso não sejam, encerra o processo retornando mensagens de erro e status HTTP 400. Além disso, foram criados contratos específicos que servem como base para os demais. Entende-se como contratos as classes que contêm os dados da requisição e a resposta e seus dados principais de uma determinada operação, sendo, para o presente trabalho, as classes com os sufixos *Request*, *Response* e *ResponseData*.

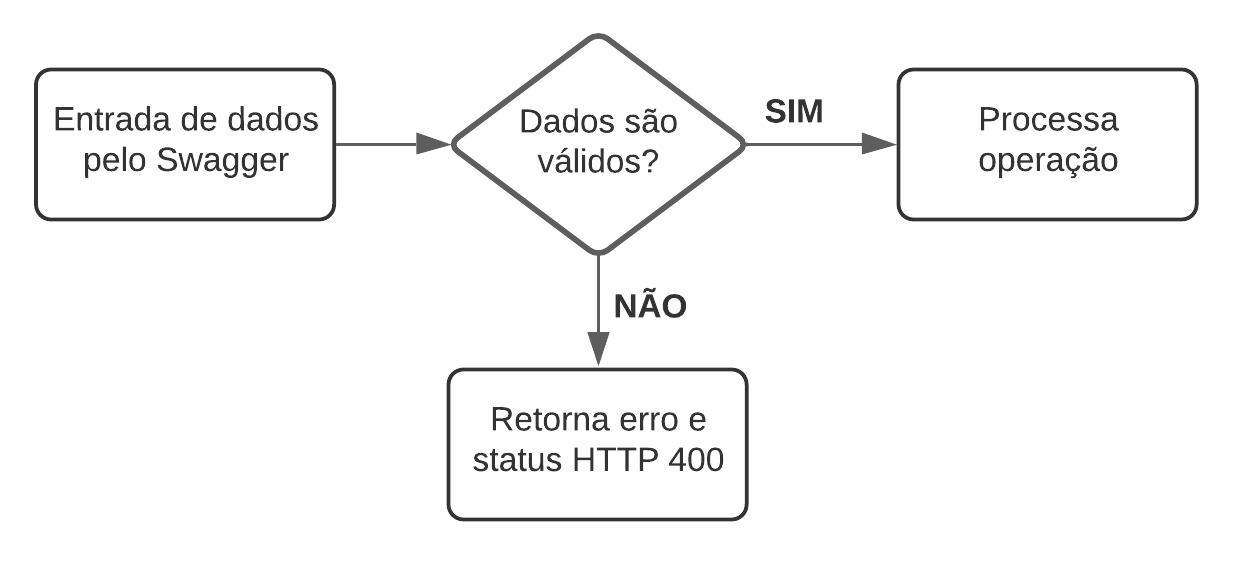


Figura 7 – Fluxograma básico das operações

Para tornar a aplicação otimizada e rápida, foi utilizado o conceito de paralelismo e assincronismo, otimizando o tempo gasto. O paralelismo diz respeito a executar tarefas em paralelo, ou seja, mais de uma tarefa será executada ao mesmo tempo. O assincronismo representa executar *threads* separadas, isto é, rodar tarefas em ordens de processamento diferentes. Esses conceitos foram aplicados usando a classe *Task* (*Microsoft*, 2021), nativa do C#, que possui todos os recursos necessários para esse gerenciamento. Isso foi utilizado em duas partes do código, sendo estas: na iteração dos dados de requisição e no cálculo de cada parâmetro dos modelos. Estas duas últimas serão detalhadas no decorrer deste trabalho. O assincronismo e o paralelismo devem ser utilizados com cautela, visto que há um maior consumo de recursos computacionais, logo, deve ser feito uma análise de custo-benefício levando em consideração, principalmente, a quantidade de tarefas em paralelo ou assíncronas que devem ser executadas.

Por último, o conceito de herança precisa ser detalhado, já é utilizado por todo o código, além de ser um tópico intrínseco à programação orientada a objetos. Segundo (DE CAMARGO, 2010), “a herança possibilita que as classes compartilhem seus atributos, métodos e outros membros da classe entre si”, isto é, ao definir que uma classe herda da outra, a classe filho ou classe derivada recebe as características da classe pai ou classe base, tendo acesso a estas e podendo manipulá-las. Isso é bastante utilizado com o intuito de evitar replicações de código, além de impor que, em determinados contextos, alguns comportamentos sejam implementados, como será melhor apresentado mais à frente no capítulo sobre a implementação numérica dos modelos de viscoelasticidade.

1. RECURSOS implementados

Para que fosse possível atingir o objetivo deste trabalho, diversos recursos tiveram que ser implementados, sendo estes separados por tópicos, conforme serão mostrados a seguir.

Para cada modelo de viscoelasticidade, foi necessário implementar duas operações e as equações constitutivas, sendo estas últimas implementadas em classes separadas com o nome de cada modelo. A primeira operação é responsável pelo cálculo das propriedades mecânicas do tecido mole, em que o código recebe e retorna parâmetros específicos de cada modelo, funcionando somente como um orquestrador, em que primeiro verifica se os dados passados são válidos e, depois, chama o método responsável por efetuar os cálculos da classe que contém as equações do modelo, além de escrever esses resultados em um arquivo. A segunda operação é responsável por fazer uma análise de sensibilidade para cada parâmetro do modelo, baseando-se na funcionalidade citada anteriormente, diferindo somente nos dados de entrada, em que é possível passar um intervalo de valores para serem testados e comparados os resultados.

## métodos numéricos

Conforme será mostrado a seguir, as equações para os modelos quase-linear e não linear possuem integrais e derivadas que precisam ser resolvidas numericamente. Posto isto, foi implementado uma classe que contém os métodos para calcular uma determinada integral e outra classe que contém as lógicas para efetuar a derivada.

O método de diferenças centrais foi utilizado para a implementação da derivada numérica por possuir uma boa precisão numérica e ser de fácil implementação, cuja formulação está mostrada pela equação 9, em que o *h* é representa um delta da variável *x*.

|  |  |
| --- | --- |
|  | (9) |

A integração numérica utilizada foi o método de Simpson composto (REGRA DE SIMPSON, 2021 e REGRAS COMPOSTAS, 2021) por poder apresentar uma grande precisão se o intervalo de integração for dividido em espaços muito pequenos. Esse método é aplicado considerando o intervalo [a, b], sendo *a* e *b* quaisquer valores reais e que a função f(x) a ser integrada seja contínua dentro deste intervalo. E é adotado *N* divisões, com  e N/2 subintervalos do tipo , com k = 0, 1,..., N/2. Assim, considerando os subintervalos, temos:

|  |  |
| --- | --- |
|  | (10) |

Aplicando a regra de Simpson simples na integral da direita do subintervalo, pode-se reescrever a equação 11:

|  |  |
| --- | --- |
|  | (11) |

Simplificando a equação 11, é possível obter a equação final para a Regra de Simpson Composta.

|  |  |
| --- | --- |
|  | (12) |

Conforme pode ser observado, para calcular a integral e derivada de uma função no código seria necessário calcular previamente todos os pontos que o método utilizaria. Para evitar que isto fosse feito, foi utilizado a classe Func, nativa do C#, que permite encapsular um método do código em uma variável, sem perder suas características.

## dados experimentais

As operações implementadas para os dados experimentais, obtidos em ensaios experimentais de ligamentos de joelho de porcos, foram duas, baseando-se em duas necessidades: remover pontos indesejáveis e extrapolar os dados.

A primeira é responsável por somente analisar os dados experimentais removendo os pontos considerados inválidos. Isso foi feito se baseando no comportamento esperado das curvas, em que, para o tempo de rampa, é esperado que a tensão aumente, logo, primeira derivada positiva e a concavidade da curva seja para cima, por tanto, segunda derivada positiva, e para a relaxação, a tensão deve diminuir, primeira derivada negativa, e, assim como na anterior, concavidade da curva para cima.

A principal função da segunda operação é extrapolar os dados através do menor valor encontrado para a segunda derivada, que, com isto, prevê a primeira derivada e, em seguida, prevê o valor da tensão para aquele ponto usando, em ambos os casos, método de integração trapezoidal. Contudo, para que isso seja possível, antes da etapa principal, os pontos são analisados utilizando a primeira operação supracitada.

## operações auxiliares

Além das operações supracitadas, algumas outras foram necessárias para automatizar algumas etapas feitas manualmente, sendo estas: uma para diminuir a quantidade de pontos em um arquivo; uma para buscar arquivos com os resultados de alguma operação nas pastas onde estes são gravados; e outra para baixar um determinado arquivo desejado. Por se tratar de uma Web API, essas duas últimas operações foram desenvolvidas para permitir que o usuário da API possa acessar e baixar quaisquer arquivos previamente gerados por este ou por outrem.

1. modelo linear de viscoleasticidade

Uma função linear é caracterizada por depender somente de uma variável. Posto isto, o modelo de viscoelasticidade linear precisa ter uma função relaxação e uma função de fluência lineares, em que ambas dependam somente do tempo. Modelos lineares são mais comumente utilizados para metais, entretanto, devido a sua simplicidade, será utilizado neste trabalho para iniciar os estudos sobre implementação numérica de modelos de viscoelasticidade.

## maxwell

O modelo de viscoelasticidade linear de Maxwell considera que o tecido mole seria representado por um amortecedor e uma mola em série, conforme mostrado na Fig. 8.

Diagrama

Descrição gerada automaticamente

Figura 8 – Representação de mola e amortecedor em série (Fung, 1993)

Observando a Fig. 8, tem-se que *F* é a força, que dividida por uma área considerada constante, se torna tensão, µ é o módulo de elasticidade e η é a viscosidade. O deslocamento do amortecedor e da mola são representados, respectivamente, por e . Usando as expressões de elasticidade, viscosidade e equilíbrio de forças e assumindo que as áreas são constantes por todo o comprimento, tem-se:

|  |  |
| --- | --- |
|  | (13) |
|  | (14) |
|  | (15) |
|  | (16) |

Derivando as equações 13 e 16, onde o ponto representa a derivada no tempo, tem-se:

|  |  |
| --- | --- |
|  | (17) |
|  | (18) |

Substituindo as equações 14, 15 e 17 em 18, tem-se:

|  |  |
| --- | --- |
|  | (19) |

Além disso, para este modelo, considera-se como deformação inicial o valor da tensão inicial dividida pelo módulo de elasticidade, conforme visto na equação 20.

|  |  |
| --- | --- |
|  | (20) |

Integrando a equação 14 em relação ao tempo, considerando a tensão constante e isolando , tem-se:

|  |  |
| --- | --- |
|  | (21) |

Isolando na equação 13 e aplicando isto e a equação 21 na equação 16, tem-se:

Substituindo e por , conforme a equação 15, e isolando , tem-se:

|  |  |
| --- | --- |
|  | (22) |

Por fim, se baseando na equação 1, a função fluência pode ser expressa conforme mostrado na equação 23.:

|  |  |
| --- | --- |
|  | (23) |

Para a relaxação, o modelo de Maxwell assume a taxa de deformação igual a zero na equação 19, possuindo uma equação diferencial de primeira ordem que possui solução, considerando e . Sua resolução está descrita em sequência.

|  |  |
| --- | --- |
|  | (24) |

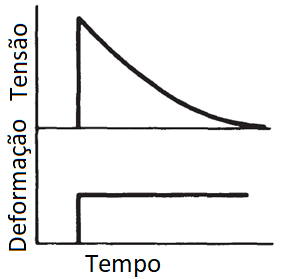
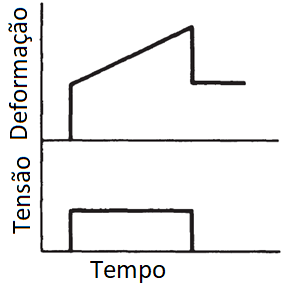
Isolando a tensão inicial da equação 20 e a substituindo em 24, tem-se:

|  |  |
| --- | --- |
|  | (25) |

Comparando-se a equação 25 com a equação 2, podemos definir a função relaxação como:

|  |  |
| --- | --- |
|  | (26) |

As curvas de relaxação e fluência para o modelo de Maxwell podem ser mais bem observados nas Fig. 9.a e 9.b, onde pode-se notar o comportamento da tensão e deformação ao longo do tempo.



(a) (b)

Figura 9 – Força e alongamento ao longo do tempo para (a) relaxação e para (b) fluência

## maxwell generalizado

O modelo generalizado de Maxwell propõe que o tecido mole pode ser representado como um conjunto de várias molas e amortecedores em série e todos estes conjuntos em paralelo com uma mola, conforme pode ser visto na Fig. 10.

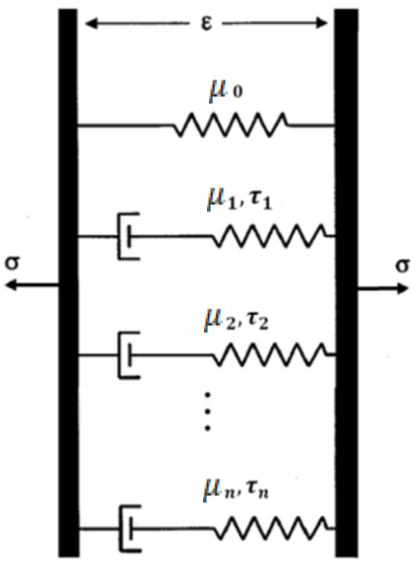


Figura 10 – Modelo generalizado de Maxwell (BABAEI, et al, 2015) com nomenclatura adaptada

Este modelo generalizado é capaz de expressar o comportamento de um tecido mole com uma grande quantidade de parâmetros, já que possui mais elementos do que o modelo simples de Maxwell. Mediante isso, a função relaxação para este modelo pode ser expressa conforme a equação 27.

|  |  |
| --- | --- |
|  | (27) |

Neste caso, a função relaxação também é chamada de série de Prony para relaxação, onde é o módulo de relaxação ou módulo de Young para a situação de equilíbrio, são os múltiplos módulos de Young para cada elemento de mola e são os tempos de relaxação de cada elemento. Conforme mostrado por (BABAEI, et al, 2015), na prática, essa equação apresenta boa aproximação utilizando somente 3 elementos no somatório.

1. modelo quase-linear de viscoelasticidade

O modelo de Viscoelasticidade Quase-Linear, proposto inicialmente por (Fung, 1993), aplica a não-linearidade da relação entre tensão e deformação expressando a tensão em duas partes: a função relaxação reduzida, que depende somente do tempo, e a resposta elástica, que depende da deformação, em que as equações utilizadas para cada parâmetro são mostradas a seguir.

Para facilitar os cálculos, este último pode ser expresso dependendo somente do tempo, já que a deformação depende deste parâmetro. Além disso, neste modelo é utilizado uma formulação baseada na equação 8 para o cálculo da tensão, fazendo uma consideração análoga, porém usando as duas parcelas citadas acima.

Este é um modelo comumente utilizado para tecidos moles, por conseguir representar a realidade com boa aproximação conforme será mostrado mais a diante nas comparações entre os resultados experimentais e numéricos. As constantes necessárias para as equações não apresentam grandes dificuldades de serem obtidas, já que não são necessários muitos experimentos para tal. Possuindo os dados experimentais, basta aplicar ajustes de curva através de softwares como o Origin para calculá-las. Entretanto, assim como qualquer modelo, este apresenta limitações, uma vez que para diferentes relaxações e patamares de deformação são encontrados valores diferentes para estas constantes.

Ademais, para este trabalho, são feitas duas ponderações relacionadas ao tempo de rampa: considerar e desconsiderar. Ao considerar o tempo de rampa, é possível considerar diversas relaxações, pela presença das variáveis *A* e *B* mostradas nos cálculos da resposta elástica que só são possíveis de serem obtidas ao considerar este tempo. Para este projeto foi considerado somente duas relaxações.

## Equacionamento

(Fung, 1993) propõe equações para serem usadas para o cálculo da resposta elástica, função relaxação reduzida e tensão para somente uma relaxação. Contudo, como será considerado duas relaxações, faz-se necessário uma reformulação destas equações conforme será apresentada em seguida.

### Deformação

Para a descrição da deformação, é utilizado a equação 28 quando procura-se descrever como é feito experimentalmente no laboratório, e a equação 29 quando o tempo de rampa não é considerado. Os parâmetros e representam, respectivamente, a taxa de deformação aplicada durante o experimento, sendo para o caso de aumento na deformação e para o caso de diminuição da deformação, e o tempo de rampa, além disso e representam os tempos em que a deformação é alterada, seja aumentando, diminuindo ou ficando constante.

|  |  |
| --- | --- |
|  | (28) |
|  | (29) |

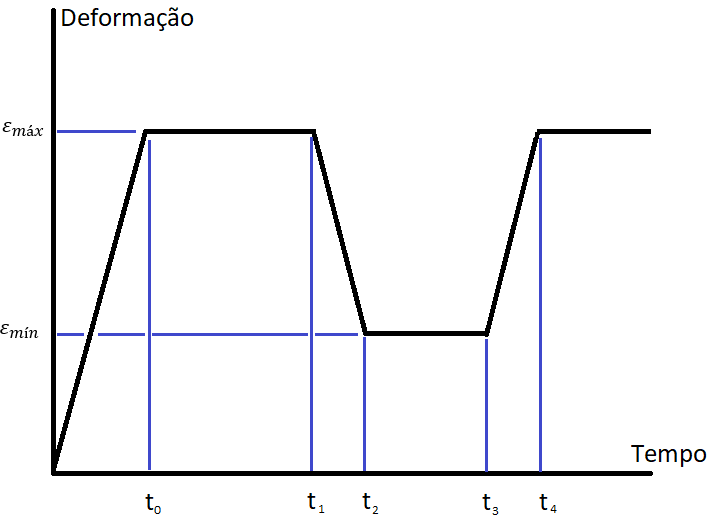


Figura 11 – Gráfico de deformação por tempo para duas relaxações

Desse modo, pode-se calcular a derivada da deformação, que mais a frente será usada para calcular a tensão. Derivando a equação 29, tem-se:

|  |  |
| --- | --- |
|  | (30) |

Sendo assim, a derivada somente será diferente de zero quando a deformação não for constante.

### Resposta elástica (parcela elástica)

A resposta elástica pode ser calculada através da equação 31, sendo em função da deformação, porém, como supracitado, esta é em função do tempo, a resposta elástica poderia ser reescrita também em função somente do tempo. Os parâmetros *A* e *B* da equação representam, respectivamente, *elastic stress constant* e *elastic stress power*. Vale salientar que mesmo que essa propriedade dependa da deformação, que para diferentes intervalos de tempo sejam usadas diferentes equações, pode-se usar somente a equação 31, além disso, observando a equação 28, pode-se perceber que em alguns intervalos de tempo a resposta elástica será constante e não dependerá do tempo.

|  |  |
| --- | --- |
|  | (31) |

Além disso, para o caso em que o tempo de rampa é desconsiderado, é usado o valor de tensão inicial aplicado ao experimento, , por todo o domínio de tempo.

|  |  |
| --- | --- |
|  | (32) |

Assim como feito para a deformação, também será calculado a derivada da resposta elástica, devido ao mesmo motivo, sua importância no cálculo da tensão, sendo assim, derivando-se a equação 31.

|  |  |
| --- | --- |
|  | (33) |

Para esta equação, vale a mesma consideração feita para a 31, onde é válido utilizar, para fins computacionais, a equação 33. Porém, vale relembrar que só é diferente de zero para os casos em que a deformação não é constante, sendo assim, é somente nesses momentos em que a derivada da resposta elástica é diferente de zero.

### Função relaxação reduzida (parcela viscosa)

A função relaxação reduzida corresponde a parcela viscosa das equações de viscoelasticidade, podendo ser escrita de duas maneiras segundo (Fung, 1993), a primeira, equação 34, é chamada de simplificada por apresentar maior facilidade de ser obtida em experimentos por ser representada por uma séria de somas de exponenciais, enquanto a segunda, equação 35, foi obtida a partir de diversos cálculos matemáticos mais complexos baseados no modelo de Kelvin (*standard linear solid*) (Fung, 1993). Vale ressaltar que para o presente trabalho será considerado que a equação 36 é composta por somente três somas de exponenciais, conforme citado anteriormente para o modelo de Maxwell generalizado. Ademais, vale lembrar que para o tempo igual a zero, a função relaxação reduzida é igual a 1, .

Os parâmetros *C* e correspondem, respectivamente, a amplitude de relaxação e o tempo de relaxação, para a equação 34, os índices são somente para diferenciar cada constante, enquanto para a equação 35, representa o tempo de relaxação rápido e , o tempo de relaxação lento, sendo sempre maior do que .

|  |  |
| --- | --- |
|  | (34) |
|  | (35) |

Como , pode-se afirmar que , sendo assim,  pode ser reescrito como:

Além disso, temos que:

Logo:

Por fim, a equação 35 pode ser reescrita como:

|  |  |
| --- | --- |
|  | (36) |

Simplificando, tem-se:

|  |  |
| --- | --- |
|  | (37) |

Como feito nas propriedades anteriores, calcula-se a derivada da função relaxação reduzida para as equações 34 e 37.

Derivando-se a equação 34:

|  |  |
| --- | --- |
|  | (38) |

Derivando-se a equação 37:

Aplicando a definição de cálculo para derivada de uma integral determinada:

Em que:

Tem-se que:

Logo:

Portanto:

|  |  |
| --- | --- |
|  | (39) |

### Tensão

Para se calcular a tensão existem 3 diferentes formas de acordo com (Fung, 1993), sendo estas expressas abaixo.

|  |  |
| --- | --- |
|  | (40) |
|  | (41) |
|  | (42) |

Conforme mostrado anteriormente, a resposta elástica e a função relaxação reduzida dependem somente do tempo, o que permite reescrever as derivadas parciais como derivadas totais. Ademais, tem-se que e .

|  |  |
| --- | --- |
|  | (43) |
|  | (44) |
|  | (45) |

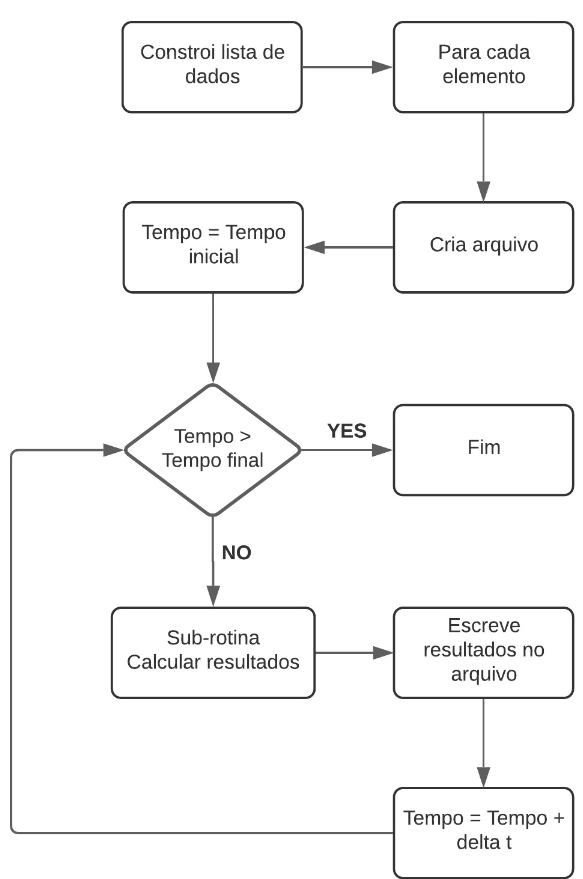
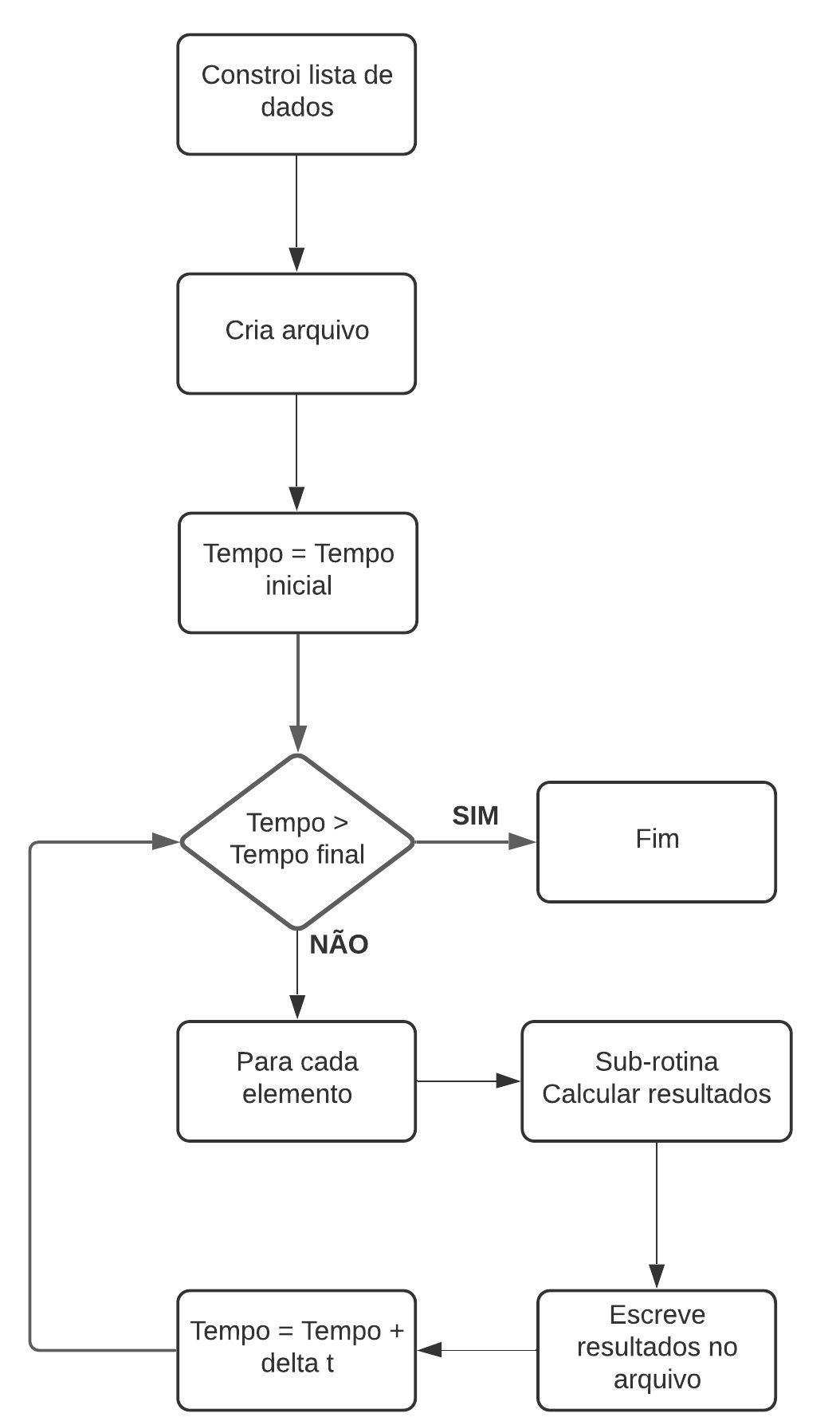
Segundo (Fung, 1993), essas equações são matematicamente equivalentes, logo, em teoria, devem retornar os mesmos valores ou resultados muito próximos. Entretanto, vale ressaltar que a equação 40 é amplamente utilizada em diversos outros trabalhos, sendo considerada a principal para este estudo e utilizada como base para comparação dos resultados obtidos nas demais equações. Conforme será apresentado na parte de implementação numérica, a utilização destas equações sem quaisquer tratamentos não apresenta viabilidade por impactar negativamente no tempo de execução do código, sendo necessário reescrevê-las com o objetivo de diminuir otimizar o intervalo de tempo da integral. Isso está descrito em detalhes do Anexo *A* deste documento.

1. implementação numérica

A implementação numérica de todos os modelos apresentados neste trabalho foi feita em duas partes: criação de uma classe que contém as equações constitutivas do modelo, sendo chamada de modelo neste capítulo para facilitar sua citação, e criação das operações para efetuar os cálculos necessários. Isso foi com o objetivo de evitar repetições de código e torná-lo mais legível e de fácil manutenção, de modo a atender os padrões de desenvolvimento de software citados na introdução deste documento, além de se basear no conceito de responsabilidade única, um dos princípios SOLID para programação orientada a objetos, em que cada classe deve ter somente uma responsabilidade.

Durante o desenvolvimento, percebeu-se que há parâmetros em comum que precisam ser calculados, dessa forma, para implementar as equações constitutivas dos modelos, foram criadas abstrações que definem o que cada modelo precisa implementar. Posto isto, no código, pode-se observar separação entre os modelos linear e quase-linear, em que cada um possui uma classe abstrata própria, além de ter sido criada uma classe abstrata que estas herdam que define os métodos que todos os modelos de viscoelasticidade devem possuir, sendo um para calcular cada um dos seguintes parâmetros a deformação, a tensão, as condições iniciais e o resultado.

Além disso, as operações que calculam os resultados e fazem a análise de sensibilidade funcionam da mesma maneira para cada modelo, com fluxograma mostrado nas figuras 12.a e 12.b. As únicas mudanças nestas são para definir quais dados de entrada devem ser utilizados, alterando a forma como é construída a lista de dados, e qual interface para o modelo deve ser utilizada, alterando as contas realizadas dentro da sub-rotina “Calcular resultados”.

(a) (b)

Figura 12 – Fluxograma das operações de (a) calcular os resultados do modelo e (b) fazer análise de sensibilidade das variáveis

Vale lembrar, que, conforme pode ser observado nos fluxogramas da Fig. 12, os dados de resposta são retornados da mesma forma, em um arquivo de formato CSV, para que este possa ser utilizado em algum outro software de geração gráfica, como o Origin.

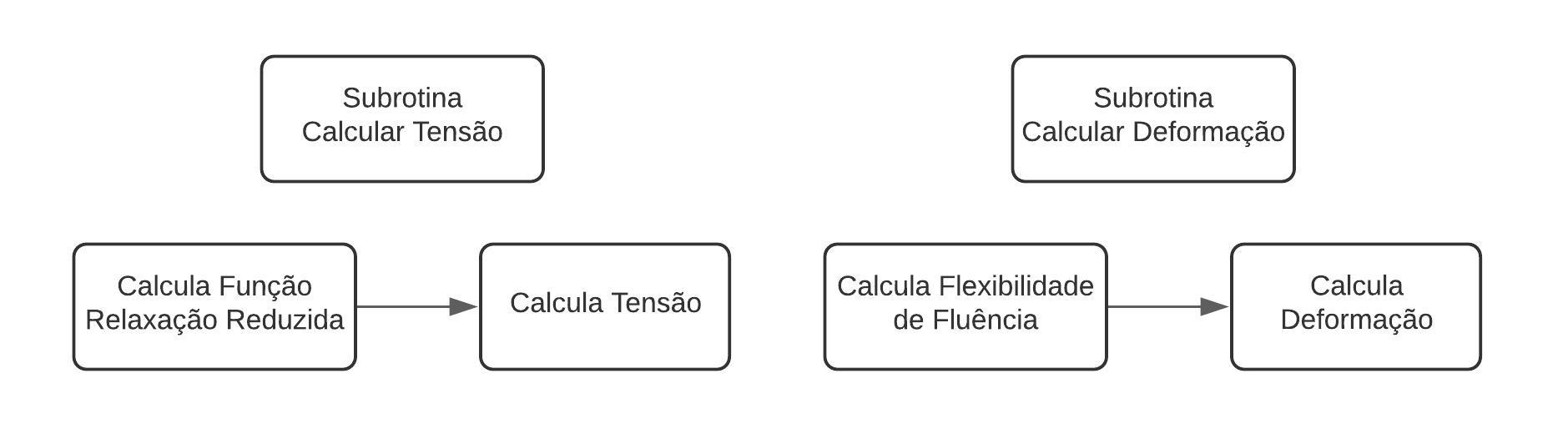
Conforme pode ser visto nos fluxogramas da Fig. 12, é necessário construir a lista de dados de entrada a ser utilizada para efetuar os cálculos na sub-rotina, também vista nos fluxogramas. Isso foi feito para seguir o princípio de responsabilidade única apresentado pelo padrão SOLID, já que os contratos das operações devem ser manipulados somente por esta e classes externas não devem ter conhecimento sobre estes contratos. Posto isso, fez-se necessário criar classes que atuariam como contratos para os modelos, em que existe uma responsável por conter os dados de entrada e outra, os resultados dos cálculos. Desta forma, pode-se observar o motivo da etapa de construir a lista de dados, em que os dados de requisição da operação são mapeados para o contrato do modelo. Mesmo isso conferindo maior complexidade ao código, essa decisão facilitou as manutenções necessárias no código, além de ter permitido escalabilidade do código e adição de recursos com grande facilidade. Vale salientar que a iteração dentro desta lista é feita de maneira assíncrona usando a classe *Task*, nativa do C#, para reduzir o tempo de execução, visto que, antes dessa alteração, em casos extremos chegou a durar horas e o tempo médio era na faixa de minutos, enquanto, com essa mudança, os tempos máximos de execução passaram a ser na ordem de segundos e o tempo médio na ordem de décimos de segundo.

Em cada operação os dados de saída são escritos de maneiras diferentes no arquivo. Para a operação que gera os resultados de um determinado modelo de viscoelasticidade, os resultados são escritos em arquivos separados para cada elemento dentro da lista criada inicialmente, já para a operação que faz análise de sensibilidade das variáveis, todos os resultados são escritos no mesmo arquivo, sendo escrito em colunas diferentes para dados de entrada diferentes. Isso é feito pois o objetivo principal desta última é a comparação de resultados.

A sub-rotina “Calcula resultados” está implementada na operação em uma única linha em que é feita uma chamada ao método do modelo que calcula os resultados, deixando todas as lógicas de cálculos dentro do modelo, conforme citado anteriormente.

## modelo linear

Para os modelos lineares de viscoelasticidade, os dados de entrada possuem a viscosidade e a rigidez do material, além da tensão inicial, quando deseja calcular a deformação, e da deformação inicial, quando quer calcular a tensão. O fluxograma para as sub-rotinas para esses cálculos pode ser visto na figura 13.



(a) (b)

Figura 13 – Fluxograma para as sub-rotinas que calculam: (a) a tensão e (b) a deformação.

Os modelos lineares possuem dois métodos específicos que precisam ser implementados, que são para calcular função relaxação e outro para calcular a função fluência. Vale salientar que, como é utilizado a herança entre as classes, um determinado modelo linear só precisa implementar o que for específico a este, logo, pode-se utilizar as lógicas presentes na classe mais genérica. Os dados de resposta para este modelo são função fluência, deformação, função relaxação e tensão.

## modelo quase-linear

Os modelos quase-linear apresentam maior complexidade na estruturação dos dados de entrada, uma vez que, para cada consideração feita, é passado parâmetros diferentes, logo, a etapa de construir a lista de dados, visto nas figuras 12.a e 12.b, é alterada para cada consideração para que seja possível mapear esses valores corretamente.

Ao considerar o tempo de rampa, é necessário passar na requisição da operação as constantes *A* e *B* para que seja possível calcular a resposta elástica, entretanto, ao desconsiderar, é necessário informar somente a tensão inicial, em que será considerado que este é o valor da resposta elástica e esta é constante por todo o domínio de tempo. Como para calcular a função relaxação reduzida há duas equações (34 e 37) e cada uma apresenta constantes distintas que são necessárias, foi feito contratos distintos para cada uma. Dessa forma, os possíveis contratos são uma combinação dessas considerações.

Os modelos quase-lineares possuem diversos métodos específicos, sendo um para calcular cada um dos seguintes parâmetros: derivada da deformação; resposta elástica e sua derivada; função relaxação reduzida e sua derivada; e tensão, sendo um método para cada equação apresentada anteriormente (40, 41 e 42). No decorrer desta implementação, percebeu-se que as equações utilizadas para a tensão precisavam ser otimizadas com o objetivo de reduzir o tempo de execução. A primeira solução adotada foi utilizar o conceito de paralelismo, executando em paralelo os cálculos dos parâmetros que compõem os resultados do modelo, sendo estes: deformação, função relaxação reduzida, resposta elástica e as 3 formas de calcular a tensão. Isso gerou uma grande redução no tempo de execução aliado à outra alteração que aplica assincronismo ao código citada anteriormente. Sendo assim, o fluxograma para a sub-rotina que calcula os resultados pode ser visto na Fig. 14.

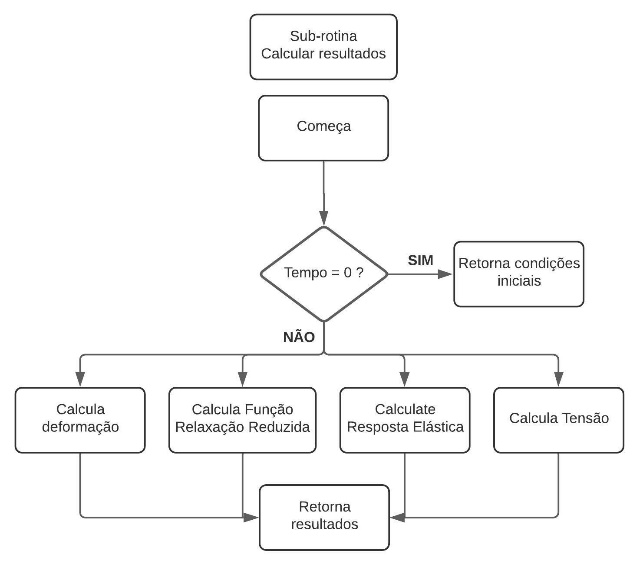


Figura 14 – Fluxograma da sub-rotina que calcula os resultados para modelo Quase-Linear

A segunda solução adotada foi reescrever as equações utilizadas com objetivo de encontrar equações específicas para cada intervalo de tempo, conforme mostrado no Anexo *A* deste documento. Estas propostas de fato auxiliaram e atingiram seu objetivo, entretanto, para a segunda solução, inicialmente eram utilizadas vários *ifs* no código o que o tornava difícil de manutenção e prejudicava a compreensão. Para sanar este problema, aplicou-se o *design pattern* *Strategy*, que, segundo (Refactoring Guru), “é um padrão de projeto comportamental que permite que você defina uma família de algoritmos, coloque-os em classes separadas, e faça os objetos deles intercambiáveis”, isto é, é um padrão de projeto que permite a segmentação do código de modo a ter partes separadas com lógicas diferentes e que são escolhidas baseadas no contexto. Sendo assim, foi desenvolvido uma classe para cada intervalo de tempo contendo as equações específicas deste período, além de um orquestrador, que baseado no tempo decide qual classe deve ser utilizada. Vale lembrar que todo o código utilizado para este trabalho se encontra no GitHub, no repositório referenciado neste documento (Silveira, 2020).

REFERÊNCIAS

Fung, Y. **Biomechanics: Mechanical Properties of Living Tissues.** 1993. Springer New York, University of Michigan.

Refactoring Guru. **Strategy.** Disponível em: https://refactoring.guru/pt-br/design-patterns/strategy. Acessado em 27 abr. 2021.

SILVEIRA, Bruno M. **SoftTissue, Repositório do GitHub.** 2020. Disponível em: https://github.com/M3110/SoftTissue. Acessado em: 24 maio 2021.

DE BARROS, Stephanie A. S. **Research on viscoelasticity models for human knee ligaments.** Relatório final de iniciação científica. Rio de Janeiro. 2020.

NAVARRO, R. F. **Modelos Viscoelásticos Aplicáveis a Materiais Reais:** uma Revisão. 2017. Universidade Federal de Campina Grande, Unidade Acadêmica de Engenharia de Materiais, Laboratório CERTBIO. Bodocongó, Campina Grande, PB.

ROSSETTO, N. P. **A viscosidade no alongamento de tendões.** 2009. Trabalho de conclusão de curso em Educação Física – Universidade Estadual de Campinas. Campinas, 2009.

BERNARDES, C., et al. **Determinação de parâmetros biomecânicos para o modelamento da articulação do joelho.** 2005. Laboratório de Pesquisa do Exercício, Universidade Federal do Rio Grande do Sul. Porto Alegre, 2005.

TARECO, M. A. C. **Conceitos de viscoelasticidade na modelação da fluência em estruturas mistas aço-betão.** 2014 154f. Dissertação (Mestrado) – Engenharia Civil, Faculdade de Ciências e Tecnologia. Lisboa, 2014. Disponível em: https://run.unl.pt/bitstream/10362/12481/1/Tareco\_2014.pdf. Acessado em 4 maio 2021.

QUEIROZ, José Aparecido Silva de. **Análise de estruturas flexíveis com aplicação de materiais viscoelásticos.** 2008. 101 f. Dissertação (mestrado) - Universidade Estadual Paulista, Faculdade de Engenharia de Bauru, 2008. Disponível em: https://repositorioslatinoamericanos.uchile.cl/handle/2250/2568006. Acessado em 4 maio 2021.

WEISS, Jeffrey A. e GARDINER, John C. **Computational Modeling of Ligament Mechanics.** 2001. University of Utah, Department of Bioengineering, Salt Lake City, Utah 84112.

DUENWALD, S. E.; JR., R. V.; LAKES, R. S. **Viscoelastic Relaxation and Recovery of Tendon.** 2009. University of Wisconsin-Madison. Madison, USA.

BONIFASI-LISTA, Carlos; LAKE, Spencer P.; SMALL, Michael S.; WEISS, Jeffrey A. **Viscoelastic properties of the human medial collateral ligament under longitudinal, transverse and shear loading.** 2004. University of Utah. Salt Lake City, Utah, USA.

PIAZZA, Stephen J.; DELP, Scott L. Three-Dimensional Dynamic Simulation of Total Knee Replacement Motion During a Step-Up Task. JOURNAL OF BIOMECHANICAL ENGINEERING. 2001. **ASME.** Vol. 123. P. 599-606. DOI: 10.1115/1.1406950.

ABRAMOWITCH, Steven D. e WOO, Savio L.-Y. An Improved Method to Analyze the Stress Relaxation of Ligaments Following a Finite Ramp Time Based on the Quasi-Linear Viscoelastic Theory. JOURNAL OF BIOMECHANICAL ENGINEERING. 2004. **ASME.** Vol. 126. P. 92-97. DOI: 10.1115/1.1645528.

DE PASCALIS, R.; ABRAHAMS, I.D.; PARNELL, W.J.. **On nonlinear viscoelastic deformations: a reappraisal of Fung’s quasi-linear viscoelastic model.** 2014. Proc. R. Soc. A 470: 20140058. Disponível em: http://dx.doi.org/10.1098/rspa.2014.0058. Acessado em 4 maio 2021.

DUENWALD, et al. **Constitutive equations for ligament and Other soft tissue: evaluation by experiment.** 2009. University of Wisconsin. Madison, Wisconsin, USA. DOI: 10.1007/s00707-009-0161-8.

WEINEROWSKA-BORDS, Katarzyna. Alternative Approach to Convolution Term of Viscoelasticity in Equations of Unsteady Pipe Flow. JOURNAL OF BIOMECHANICAL ENGINEERING. 2015. **ASME.** Vol. 137. p. 054501-1 até 054501-8. DOI: 10.1115/1.4029573.

JAMISON, C. E.; MARANGONI, R. D.; GLASER, A. A. **Viscoelastic properties of soft tissue by discrete model characterization.** 1968. Journal of Engineering for Industry.

MILLER, Karol. **Constitutive model of brain tissue suitable for finite element analysis of surgical procedures.** 1998. Department of Mechanical and Materials Engineering, The University of Western Australia, Nedlands/Perth WA 6907, Australia.

DEBSKI, R. E; ABRAMOWITCH, S. D.; WOO, S. L.-Y.; CLINEFF, T. D. **An evaluation of the quasi-linear viscoelastic properties of the healing medial collateral ligament in a goat model.** 2004. Annals of Biomedical Engineering.

ZHENG, N.; et al. **An analytical model of knee for estimation of internal forces during exercise.** 1998. American Sports Medicine Institute. Birmingham, Alabama, USA.

COMPOSITE CONSTRUCTION. 2020. Disponível em: http://fgg-web.fgg.uni-lj.si/~/pmoze/ESDEP/master/wg10/l0520.htm. Acessado em: 6 maio 2021.

REGRA DE SIMPSON. Instituto Superior Técnico de Lisboa. Disponível em: https://www.math.tecnico.ulisboa.pt/~calves/courses/integra/capiii33.html#:~:text=Regra%20de%20Simpson%20aplicada%20a%20dois%20sub%2Dintervalos.&text=Assim%2C%20podemos%20considerar%20tr%C3%AAs%20n%C3%B3s,cada%20um%20destes%20sub%2Dintervalos. Acessado em: 18 maio 2021.

REGRAS COMPOSTAS. Universidade Federal do Rio Grande do Sul. Disponível em: https://www.ufrgs.br/reamat/CalculoNumerico/livro-oct/in-regras\_compostas.html. Acessado em 18 maio 2021.

BABAEI, Behzad, et al. **Efficient and optimized identification of generalized Maxwell viscoelastic relaxation spectra.** 2015 Disponível em: http://dx.doi.org/10.1016/j.jmbbm.2015.10.008. Acessado em: 6 maio 2021.

XU, Qinwu e ENGQUIST, Björn. **A mathematical model for fitting and predicting relaxation modulus and simulating viscoelastic responses.** 2018. Disponível em: https://doi.org/10.6084/m9.figshare.c.4088969. Acessado em: 6 maio 2021.

WISMANS, J. S. H. M. **A three-dimensional mathematical model of the human knee joint Eindhoven.** 1980. Technische Hogeschool Eindhoven. DOI: 10.6100/IR155825

TechTudo. **O que é API e para que serve? Cinco perguntas e respostas**. 2020. Disponível em: https://www.techtudo.com.br/listas/2020/06/o-que-e-api-e-para-que-serve-cinco-perguntas-e-respostas.ghtml. Acessado em: 27 abr. 2021.

Microsoft. **Um tour pela linguagem C#.** 2021. Disponível em: https://docs.microsoft.com/pt-br/dotnet/csharp/tour-of-csharp/. Acesso em 27 abr. 2021.

Microsoft. **APIS +MICROSERVICES: The ultimate guide to creating an enterprise API platform.** 2019. Disponível em: https://azure.microsoft.com/mediahandler/files/resourcefiles/apis-microservices-ebook/Azure\_API-Microservices\_eBook.pdf. Acesso em: 27 abr. 2021.

Stack Overflow. **How to export a Swagger JSON/YAML file from Swagger UI?** 2019. Disponível em: https://stackoverflow.com/questions/48525546/how-to-export-a-swagger-json-yaml-file-from-swagger-ui. Acesso em: 24 maio 2021.

SMARTBEAR. **About Swagger.** 2021. Disponível em: https://swagger.io/about/. Acessado em: 29 abr. 2021.

MILVUS. **O que é um endpoint:** Como proteger este tipo de dispositivo. 2020. Disponível em: https://milvus.com.br/o-que-e-endpoint/. Acessado em: 29 abr. 2021.

HOSTGATOR. **API Restful:** conceito, princípios e como criar. 2019. Disponível em: https://www.hostgator.com.br/blog/api-restful/. Acessado em 29 abr. 2021.

INTRODUCING JSON. Disponível em: https://www.json.org/json-en.html. Acessado em: 29 abr. 2021.

REDHAT. **O que é API REST?** Disponível em: https://www.redhat.com/pt-br/topics/api/what-is-a-rest-api. Acessado em: 29 abr. 2021.

RICK ANDERSON. **Visão geral do ASP.NET.** Microsoft. 2019. Disponível em: https://docs.microsoft.com/pt-br/aspnet/overview. Acessado em: 29 abr. 2021.

GASPAROTTO, H. M. **Introdução ao ASP.NET MVC.** DEVMEDIA. 2014. Disponível em: devmedia.com.br/introducao-ao-asp-net-mvc/31878. Acessado em: 29 abr. 2021.

RUNSCOPE. **HTTP Status Codes.** Disponível em: https://httpstatuses.com/. Acessado em: 30 abr. 2021.

MICROSOFT. **Task Classe.** 2021. Disponível em: https://docs.microsoft.com/pt-br/dotnet/api/system.threading.tasks.task?view=net-5.0. Acessado em: 4 maio 2021.

DE CAMARGO, Wellington Balbo. **Conceitos e Exemplos – Herança: Programação Orientada a Objetos – Parte 1.** DEVMEDIA. 2010**.** Disponível em:https://www.devmedia.com.br/conceitos-e-exemplos-heranca-programacao-orientada-a-objetos-parte-1/18579. Acessado em 23 maio 2021.

* 1. Descrição das equações usadas para modelo de viscoelasticidade Quase-Linear

Equações usadas para deformação:

Equações usadas para resposta elástica:

A1 Considerando tempo de rampa

A1.1

A1.1.1

|  |  |
| --- | --- |
|  | (A1) |

A1.1.2

|  |  |
| --- | --- |
|  | (A2) |

A1.1.3

|  |  |
| --- | --- |
|  | (A3) |

A1.1.4

|  |  |
| --- | --- |
|  | (A4) |

A1.1.5

|  |  |
| --- | --- |
|  | (A5) |

A1.1.6

|  |  |
| --- | --- |
|  | (A6) |

A1.1.2 e

Para estas equações, não foram encontradas vantagens em reescrevê-las por terem a parcela , que dificultou drasticamente, já que a equação utilizada para a resposta elástica não dependeria somente do limite de integração ou somente do tempo, fazendo com que, em um limite de integração arbitrário, mais de uma equação para a resposta elástica poderia ser utilizada, diferente do que ocorria a equação .

A2 Desconsiderando tempo de rampa

Ao desconsiderar o tempo de rampa, temos que a resposta elástica sempre será constante, logo, sua derivada é igual a zero.

A2.1

Para esta equação, não será possível utilizá-la já que, como a resposta elástica é constante, sua derivada será igual a zero, logo, para todo o domínio de tempo, essa equação será igual a zero.

A2.2

|  |  |
| --- | --- |
|  | (A7) |

A2.3

|  |  |
| --- | --- |
|  | (A7) |