Recomendação de Adubação e Calagem usando Lógica *Fuzzy*

Gabriel Copini Souza[1], **Leonardo de Souza da Luz**[1],
Wilson Castello Branco Neto[1]

[1]Instituto Federal de Santa Catarina (IFSC) - Campus Lages
Curso Superior de Ciência da Computação
R. Heitor Villa Lobos, 225 - São Francisco, Lages - SC, 88506-400

{gabriel.cs05,leonardo.sl19}@aluno.ifsc.edu.br

wilson.castello@ifsc.edu.br

Abstract. *This paper presents a fertilization and liming recommendation system based on fuzzy logic. The objective is to soften the rigid thresholds in the Fertilization and Liming Manual (Manual de Adubação e Calagem para os estados de Santa Catarina e Rio Grande do Sul), providing more flexible recommendations adapted to variations in soil conditions. Tests were conducted by varying clay content, phosphorus, Cation-Exchange Capacity (CEC), and potassium to generate recommendations for P_2O_5 and K_2O. The results obtained with fuzzy logic were compared with recommendations derived from the original tables in the manual. The findings indicate that fuzzy logic is a promising approach to improving the efficiency of current recommendations by producing smoother transitions between classes while remaining consistent with the manual's rules.*
Keywords: *Fuzzy logic; Recommendation system; Fertilization; Liming.*

Resumo. *Este trabalho apresenta um sistema de recomendação de adubação e calagem baseado em lógica fuzzy. O objetivo é suavizar os limites rígidos presentes nas tabelas do Manual de Adubação e Calagem para os estados de Santa Catarina e Rio Grande do Sul, oferecendo resultados mais flexíveis e adaptados às variações nas condições do solo. Foram realizados testes variando os valores de argila, fósforo, CTC e potássio, para gerar as recomendações de P_2O_5 e K_2O. Os resultados obtidos com lógica fuzzy foram comparados com as recomendações elaboradas com a aplicação das tabelas originais do referido manual. Eles permitem concluir que o uso da lógica fuzzy proporciona uma abordagem promissora para aprimorar a eficiência das recomendações realizadas atualmente, pois foi capaz de gerar transições mais suaves entre as classes, mantendo a consistência com as regras do manual.*
Palavras-chave: *Lógica fuzzy; Sistema de recomendação; Adubação; Calagem.*

1. Introdução

A agronomia é um campo em constante evolução, impulsionado tanto pelos avanços tecnológicos quanto pela crescente demanda por maior produtividade. Desde a década de 1960, a agricultura na região Sul do Brasil passou por profundas transformações com a introdução de novas tecnologias e práticas agrícolas voltadas à maximização da eficiência produtiva (Sociedade Brasileira de Ciência do Solo, 2016).

Segundo Santos e Almeida (2017), o setor enfrenta desafios significativos relacionados à sustentabilidade do solo e à eficiência dos processos produtivos, o que torna essencial a adoção de estratégias mais precisas para adubação e correção do solo, uma vez que a produtividade agrícola está diretamente atrelada à qualidade do solo e à aplicação correta de fertilizantes. O Brasil gasta US$ 25 bilhões por ano para importar 85% dos fertilizantes (Pereira e Cardoso, 2025).

Para alcançar esse equilíbrio, são utilizados diferentes métodos de cálculo, entre eles o da Capacidade de Troca de Cátions (CTC), além de tabelas predefinidas, como as presentes no Manual de Adubação e Calagem para os estados de Santa Catarina e Rio Grande do Sul (MACSCRS) da Sociedade Brasileira de Ciência do Solo (2016). Apesar de amplamente adotadas, essas tabelas apresentam limitações, sobretudo pela rigidez na consideração das variáveis ambientais e edáficas (Corrêa et al., 2015).

Dentro desse contexto, a lógica *fuzzy* surge como uma abordagem alternativa fundamentada na teoria dos conjuntos difusos, proposta por Zadeh (1965). Diferentemente da lógica clássica binária, a lógica *fuzzy* permite representar e manipular imprecisões por meio de graus de pertinência. Essa característica a torna especialmente útil em cenários como o da agricultura, nos quais variáveis como a composição do solo e as condições ambientais oscilam de forma contínua. A modelagem baseada em lógica *fuzzy* permite uma adaptação mais flexível e realista dos parâmetros, tornando o processo de recomendação agronômica mais robusto, preciso e compatível com a complexidade do ambiente agrícola. Como consequência, há potencial para ganhos significativos na produtividade e redução das despesas e dos impactos ambientais causados pelo uso inadequado de insumos.

Considerando esses desafios e oportunidades, este trabalho tem como objetivo desenvolver um sistema web para recomendação de adubação, automatizando a análise de laudos técnicos e aplicando conceitos de lógica *fuzzy* para aumentar a precisão das recomendações.

Entre os objetivos específicos do projeto, destacam-se:

- Implementar um módulo de cálculo de calagem com base no estabelecido no MACSCRS.
- Desenvolver um módulo para utilizar as tabelas do MACSCRS na realização de cálculos de adubação.
- Converter as tabelas do MACSCRS para um formato compatível com a lógica *fuzzy*.
- Integrar os componentes desenvolvidos em um sistema web.
- Comparar os resultados das abordagens clássicas e *fuzzy*.

A metodologia deste trabalho caracteriza-se como uma pesquisa de natureza aplicada, com abordagens qualitativas e quantitativas, exploratória quanto aos seus objetivos,

bibliográfica e experimental quanto aos procedimentos técnicos, sendo organizada em quatro etapas principais. Primeiramente, será realizada uma revisão bibliográfica, utilizando livros e artigos científicos, sobre os fundamentos de adubação, calagem e correção de macronutrientes, com ênfase nas práticas adotadas nos estados de Santa Catarina e Rio Grande do Sul, além de um estudo sobre os princípios e aplicações da lógica *fuzzy*. Na segunda etapa, será conduzido o desenvolvimento do sistema, utilizando as tecnologias *Python, FastAPI, Pydantic, OAuth2* com *JWT, PostgreSQL, Vue3* e *TypeScript*, contemplando a definição da arquitetura e a implementação dos módulos funcionais. Em seguida, o sistema será testado com dados reais fornecidos por agrônomos locais, viabilizando sua avaliação em situações práticas. Por fim, os resultados obtidos serão analisados com o intuito de verificar a eficácia do sistema proposto, comparando os resultados das abordagens clássicas e baseadas em lógica *fuzzy*.

Dessa forma, a proposta deste trabalho se insere na intersecção entre a agronomia e a tecnologia da informação, explorando a aplicação de métodos computacionais inteligentes para enfrentar desafios agronômicos reais. Ao unir o conhecimento técnico da adubação com a flexibilidade da lógica *fuzzy*, espera-se oferecer uma ferramenta prática, eficiente e adaptável à realidade dos profissionais do campo. Acredita-se que a integração dessas abordagens possa não apenas otimizar os processos de tomada de decisão, mas também contribuir para uma agricultura mais sustentável e orientada por dados.

Quanto à estrutura do artigo, é apresentada da seguinte forma: na Seção 1, apresenta-se a introdução ao tema, contextualizando os desafios da agricultura moderna e a proposta do sistema. A Seção 2 aborda o referencial teórico, incluindo os fundamentos da lógica *fuzzy* e uma revisão de trabalhos similares. A Seção 3 detalha o desenvolvimento do sistema, contemplando a arquitetura, os códigos implementados, a interface e a definição das regras *fuzzy*. Em seguida, a Seção 4 apresenta os resultados obtidos com a aplicação prática do sistema e a comparação entre as abordagens clássica e *fuzzy*. Por fim, a Seção 5 traz as conclusões, destacando as contribuições do trabalho e possíveis melhorias futuras.

2. Referencial teórico

A crescente demanda por sistemas agrícolas mais eficientes e sustentáveis tem impulsionado o desenvolvimento de novas abordagens no manejo da fertilidade do solo. Paralelamente, o avanço de tecnologias computacionais, como a lógica *fuzzy*, tem possibilitado a modelagem de sistemas complexos com maior flexibilidade e precisão. Neste capítulo, são apresentados os fundamentos teóricos relacionados tanto aos princípios e práticas do manejo da fertilidade do solo quanto à lógica *fuzzy*, com o objetivo de embasar a aplicação dessa técnica na interpretação de variáveis agronômicas e na tomada de decisão em ambientes incertos.

2.1. Princípios e práticas de manejo da fertilidade do solo

O solo é um recurso dinâmico e fundamental para a produção agrícola, cuja qualidade e produtividade dependem diretamente de processos naturais e de intervenções de manejo. Nesta seção, são apresentados os principais fundamentos relacionados à fertilidade do solo e às práticas agronômicas adotadas para sua manutenção e correção.

A fertilidade do solo resulta da ação conjunta de processos de intemperismo, ciclagem de nutrientes e decomposição de biomassa, gerando uma camada biologicamente

ativa que sustenta o crescimento vegetal e garante a produtividade dos agroecossistemas (Novais et al., 2007).

Segundo Novais et al. (2007, p. 43),

> solo produtivo é um solo fértil, ou seja, que contém os nutrientes essenciais em quantidades adequadas e balanceadas para o normal crescimento e desenvolvimento das plantas cultivadas e que apresenta ainda boas características físicas e biológicas, está livre de elementos tóxicos e encontra-se em local com fatores climáticos favoráveis.

A fertilidade do solo é fundamental para a manutenção da produtividade dos agroecossistemas, pois práticas que preservam e equilibram os nutrientes evitam perdas de fertilidade e garantem o bom desempenho desses sistemas, que atualmente ocupam mais de um quarto da área global.

A calagem é a prática de correção da acidez do solo por meio da aplicação de calcário, cujo principal mecanismo de ação está fundamentado na troca catiônica. Esse processo eleva o pH do solo, neutraliza íons de alumínio tóxicos, favorece a microbiota do solo, reduz a fixação excessiva de fósforo e fornece cálcio e magnésio às plantas, promovendo maior eficiência na absorção de água e nutrientes (Novais et al., 2007).

Adubação é a prática de aplicar insumos ao solo ou diretamente às plantas, sejam eles de origem mineral ou orgânica, naturais ou sintéticos, cujo objetivo é fornecer um ou mais nutrientes essenciais ao crescimento vegetal. Essa operação visa suprir as necessidades nutritivas das culturas e, assim, promover o desenvolvimento saudável das plantas (Novais et al., 2007).

Segundo Melo (2003),

> existem três tipos fundamentais de adubação: a de correção, efetuada antes do plantio, a de crescimento, realizada durante a fase de crescimento das mudas (durante os dois a três primeiros anos), e a de manutenção, realizada durante a vida produtiva da planta.

De acordo com Melo e Brunetto (2016):

- Adubação de pré-plantio: é baseada em análise de solo e tem como objetivo corrigir as carências nutricionais dos solos. Os adubos devem ser incorporados ao solo antes do plantio das mudas.
- Adubação de crescimento: como o solo foi corrigido pela adubação de correção, aqui se recomenda apenas a aplicação de N, que é o nutriente com mais baixa disponibilidade nos solos.
- Adubação de manutenção: tem como finalidade devolver ao solo, ao menos, a quantidade de nutrientes extraída pelas plantas. É uma prática realizada anualmente.

A adubação é fundamental para garantir a produtividade e a saúde das plantas, pois repõe os nutrientes que o solo perde ao longo do cultivo e evita seu empobrecimento. Ao fornecer elementos como nitrogênio, fósforo e potássio em concentrações adequadas, mantém-se o equilíbrio químico do solo, favorecendo o crescimento radicular, a absorção de água e a síntese de compostos essenciais ao metabolismo vegetal. Dessa forma, além de maximizar rendimentos, a adubação contribui para a sustentabilidade do sistema agrícola, preservando a fertilidade do solo a longo prazo (Novais et al., 2007).

2.1.1. Cálculo de calagem

Nessa seção, são apresentados os dois métodos para o cálculo de calagem mais utilizados:

- Método SMP (Shoemaker et al., 1961):
  O método SMP, desenvolvido por Shoemaker, McLean e Pratt, é uma técnica utilizada para estimar a necessidade de calagem dos solos. De acordo com Novais et al. (2007), o método SMP baseia-se na medida do decréscimo de pH de uma suspensão solo:solução (relação 1:10) em contato com uma solução-tampão de acetato de amônio $1\ \mathrm{mol\,L^{-1}}$ ajustada a $pH_{SMP} = 7{,}0$. A partir do valor de pH_{SMP} obtido, utiliza-se uma curva de neutralização ou tabelas previamente calibradas (Tabela 1) para relacionar cada nível de pH_{SMP} à dose de $\mathrm{CaCO_3}$ necessária para atingir o pH em água desejado. Sua calibração consiste em correlacionar, por incubação com $\mathrm{CaCO_3}$, o pH_{SMP} de diferentes solos às quantidades de calcário necessárias para elevar o pH a valores-alvo (por exemplo, 6,0 ou 6,5), devendo-se ajustar as tabelas ou curvas à realidade regional. Esse método é oficialmente utilizado nos estados de Santa Catarina e Rio Grande do Sul. Em Sociedade Brasileira de Ciência do Solo (2016, p. 70), há uma tabela similar, porém mais completa que a Tabela 1, com a dose de calcário a ser adicionada para todos os níveis de pH_{SMP} entre 4,4 e 7,1, variando de 0,1 em 0,1.

Tabela 1. Necessidade de calagem de solos de acordo com o pH_{SMP} (relação 10:10:5, solo, água, solução-tampão).

pH_{SMP}	NC para pH_{H_2O}		
	5,5	6,0	6,5
4,5	12,5	17,3	24,0
5,0	6,6	9,9	13,3
5,5	3,7	6,1	8,6
6,0	1,6	3,2	4,9
6,5	0,4	1,1	2,1

Fonte: Novais et al. (2007, p. 238).

- Método de Saturação por Bases:
  Segundo Sociedade Brasileira de Ciência do Solo (2016), o método de saturação por bases consiste em determinar a dose de calcário necessária para elevar a saturação de bases (V%) do solo até um valor de referência, mantendo a estimativa da acidez potencial (H + Al) pelo índice SMP. A partir dos valores de V% atual (V_2) e de capacidade de troca de cátions a pH 7,0 ($\mathrm{CTC_{pH\,7.0}}$) fornecidos na análise de solo, calcula-se a necessidade de calcário (NC, em t/ha) para corrigir os primeiros 20 cm de perfil, por meio da Equação 1 :

$$\mathrm{NC} = \frac{V_1 - V_2}{PRNT^1} \times \mathrm{CTC_{pH\,7.0}} \tag{1}$$

em que V_1 é a saturação por bases desejada (65% para pH 5,5; 75% para pH 6,0; 85% para pH 6,5), V_2 a saturação por bases do solo e $\mathrm{CTC}_{\mathrm{pH\,7.0}}$ a capacidade de troca de cátions a pH 7,0, em $\mathrm{cmol_c\,dm^{-3}}$.
O método de saturação por bases tem como vantagem a flexibilidade da recomendação de calagem para diferentes culturas. Esse método é principalmente utilizado nas regiões Norte, Nordeste, Sudeste e Centro Oeste (Novais et al., 2007).

Independentemente do método de cálculo empregado, o passo seguinte à determinação da necessidade de calcário é a escolha do tipo de calcário ou mistura de múltiplos tipos a serem utilizados. Nesta fase considera-se a quantidade de Ca e Mg que o solo já possui e também a quantidade que se encontra em cada tipo de calcário, para atingir o equilíbrio de Ca e Mg no solo.

2.1.2. Cálculos de adubação

Para os cálculos de adubação são usados manuais de recomendação. De acordo com Novais et al. (2007, p. 797-798),

> os critérios de diagnóstico da fertilidade com base na análise química do solo, assim como as orientações para fertilização das culturas, são organizados em manuais. [...] Citam-se, como exemplos: Recomendações de Adubação e Calagem para o Estado de São Paulo – Boletim Técnico 100 (Raij et al., 1996), Recomendações para o Uso de Corretivos e Fertilizantes em Minas Gerais – 5ª Aproximação (Ribeiro et al., 1999), Manual de Adubação e de Calagem para os Estados do Rio Grande do Sul e Santa Catarina (Sociedade..., 2004).

Esse projeto contempla somente a fase de adubação de pré-plantio; portanto, não são descritos os processos de adubação de crescimento e de manutenção. Toda a sequência dessa seção sobre adubação de pré-plantio está baseada em Sociedade Brasileira de Ciência do Solo (2016, p. 91-96).

A primeira etapa consiste em verificar a classe de disponibilidade de P e K do solo, utilizando as tabelas do referido manual.

Para encontrar a classe de P, é necessário conhecer a classe de teor de argila do solo por meio da Tabela 2 e o tipo de cultura, os quais estão separados em grupos. Por exemplo, o Grupo 1 contém alho e beterraba, enquanto o Grupo 2 contém culturas de frutíferas e pastagens (exceto pastagem natural). Tendo essas duas informações, basta usar a tabela que relaciona a classe do teor de argila com o respectivo grupo da cultura. A Tabela 3 apresenta esta tabela para as culturas do Grupo 2.

Considerando uma amostra de solo com teor de argila de 30 % e teor de fósforo extraído pelo método Mehlich-1 de 15 mg P/dm³. De acordo com a Tabela 2, o teor de argila de 30 % enquadra-se na faixa de 21–40 %, correspondendo à Classe 3. Em seguida, aplicando-se esta Classe de argila na Tabela 3, verifica-se que um teor de fósforo de 15 mg P/dm³ situa-se na faixa de 12,1 – 18,0 mg P/dm³, correspondente à disponibilidade *Médio*. Portanto, para este exemplo, o solo apresenta disponibilidade de fósforo média.

[1] PRNT (Poder Relativo de Neutralização Total): índice que combina o Poder de Neutralização (PN) e a Reatividade (RE) do corretivo para indicar a fração efetiva que neutralizará a acidez do solo.

Tabela 2. Interpretação dos teores de argila, matéria orgânica e capacidade de troca de cátions ($CTC_{pH7,0}$) do solo.

Argila		Matéria Orgânica		$CTC_{pH7,0}$	
Faixa (%)	Classe	Faixa (%)	Classe	Faixa (cmol/dm^3)	Classe
$\leq$ 20	4	$\leq$ 2,5	Baixo	$\leq$ 7,5	Baixa
21–40	3	2,6–5,0	Médio	7,6–15,0	Média
41–60	2	>5,0	Alto	15,1–30,0	Alta
>60	1	-	-	>30,0	Muito alta

Fonte: Sociedade Brasileira de Ciência do Solo (2016, p. 91).

Tabela 3. Interpretação do teor de fósforo no solo extraído pelo método Mehlich-1, conforme o teor de argila para culturas do Grupo 2 (culturas de grãos, exceto arroz irrigado; hortaliças, exceto as do Grupo 1; pastagens, exceto pastagem natural; frutíferas e gengibre).

Classe de disponibilidade	Classe de teor de argila[1,2]			
	1	2	3	4
	mg de P/dm^3			
Muito baixo	$\leq$3,0	$\leq$4,0	$\leq$6,0	$\leq$10,0
Baixo	3,1 – 6,0	4,1 – 8,0	6,1 – 12,0	10,1 – 20,0
Médio	6,1 – 9,0	8,1 – 12,0	12,1 – 18,0	20,1 – 30,0
Alto	9,1 – 18,0	12,1 – 24,0	18,1 – 36,0	30,1 – 60,0
Muito alto	>18,0	>24,0	>36,0	>60,0

[1] Teores de argila: classe 1 = >60%; classe 2 = 60 a 41%; classe 3 = 40 a 21%; classe 4 = $\leq$20%.
[2] Caso a análise tenha sido feita por Mehlich-3, transformar previamente os teores em "equivalentes Mehlich-1", conforme equação PM1 = PM3/(2 - (0,02 × arg)) (Capítulo 4).
Fonte: Sociedade Brasileira de Ciência do Solo (2016, p. 94).

A busca pela classe de K, segue a mesma lógica da de P, encontrando a classe do $CTC_{pH7,0}$ por meio da Tabela 2 e o tipo da cultura, que também está dividido em grupos por uma tabela. A partir desses dois valores, deve-se usar a tabela que relaciona a classe de $CTC_{pH7,0}$ com o respectivo grupo da cultura. A Tabela 4 apresenta esta tabela para as culturas do Grupo 2.

Tomando como exemplo uma amostra de solo com $CTC_{pH7,0}$ igual a 20,0 cmol /dm³ e teor de potássio extraído pelo método Mehlich-1 de 50 mg K_2O/dm³. De acordo com a Tabela 4, para solos cuja $CTC_{pH7,0}$ situa-se na faixa de 15,1–30,0 cmol /dm³, um teor de K_2O de 50 mg K_2O/dm³ enquadra-se na disponibilidade *Baixo* (41–80 mg K_2O/dm³).

Com as classes de disponibilidade de P e K encontradas, a última etapa necessária

Tabela 4. Interpretação do teor de potássio no solo extraído pelo método Mehlich-1, conforme a $CTC_{pH_{7,0}}$ do solo para culturas do Grupo 1 (alho, beterraba, cenoura, mandioquinha-salsa, tomateiro, batata, batata-doce e roseira de corte).

Classe de	**$CTC_{pH_{7,0}}$ do solo[(1)]**			
disponibilidade	$\leq$ 7,5	7,6 a 15,0	15,1 a 30,0	>30,0
Muito baixo	$\leq$20	$\leq$30	$\leq$40	$\leq$45
Baixo	21–40	31–60	41–80	46–90
Médio	41–60	61–90	81–120	91–135
Alto	61–120	91–180	121–240	136–270
Muito alto	>120	>180	>240	>270

(1) Caso a análise tenha sido feita por Mehlich-3, transformar previamente os teores em "equivalentes Mehlich-1", conforme equação KM1 = KM3 × 0,83 (Capítulo 4).
Fonte: Sociedade Brasileira de Ciência do Solo (2016, p. 96).

consiste em utilizar a tabela de recomendação da cultura desejada, para definir a quantidade necessária de P_2O_5/ha e K_2O/ha. A Tabela 5 é um exemplo desta tabela para as espécies frutíferas.

Com as classes de disponibilidade de fósforo *Médio* e potássio *Baixo* determinadas para o solo de exemplo, define-se as doses de adubação recomendadas para culturas frutíferas. De acordo com a Tabela 5, solos com teor de P disponível *Médio* devem receber 130 kg P_2O_5/ha, enquanto para potássio *Baixo* a recomendação é de 90 kg K_2O/ha. Dessa forma, para o solo em questão — $CTC_{pH7,0}$ = 20,0 cmol/dm^3; P = 15 mg P/dm^3; K = 50 mg K_2O/dm^3 — a dose a ser aplicada em pré-plantio é de 130 kg P_2O_5/ha e 90 kg K_2O/ha. Este procedimento assegura que a adubação atenda às necessidades nutricionais do solo, para a implantação da cultura.

Tabela 5. Quantidades de fósforo e potássio recomendadas em pré-plantio para as espécies frutíferas em função dos teores de P e K disponíveis no solo.

Interpretação do teor	**Nutriente[(1)]**	
de P e K no solo	**Fósforo** (kg de P_2O_5/ha)	**Potássio** (kg de K_2O/ha)
Muito baixo	250	150
Baixo	170	90
Médio	130	60
Alto	90	30
Muito alto	0	0

(1) Dependendo do tipo de solo, da espécie frutífera e do sistema de produção, essas doses podem ser aumentadas ou diminuídas a critério do técnico responsável pelo pomar.
Fonte: Sociedade Brasileira de Ciência do Solo (2016, p. 190).

2.2. Lógica *fuzzy*

A lógica *fuzzy* é uma extensão da lógica clássica, proposta por Zadeh (1965), que introduziu o conceito de graus de verdade. Enquanto a lógica booleana trabalha exclusivamente com valores binários, classificando proposições como verdadeiras ou falsas (0 ou 1), a lógica *fuzzy* permite que as proposições assumam valores intermediários entre 0 e 1. Esse modelo é particularmente vantajoso porque aproxima o raciocínio computacional da forma como os seres humanos interpretam fenômenos de forma imprecisa, utilizando termos vagos como *alto*, *quente* ou *baixo*, sem a necessidade de definições rígidas. Assim, a lógica *fuzzy* permite representar matematicamente o raciocínio aproximado, facilitando a modelagem de problemas complexos em que as fronteiras entre categorias não são claramente definidas.

Em termos de aplicação prática, a lógica *fuzzy* é indicada para situações em que há imprecisão ou subjetividade nas informações disponíveis. Em diversos cenários, as variáveis envolvidas não apresentam limites claros ou comportamentos exatos, o que torna os métodos tradicionais menos eficientes. Nesse contexto, a lógica *fuzzy* se mostra uma solução mais flexível. Exemplos de aplicações incluem o controle de sistemas de climatização, nos quais o ajuste da temperatura é realizado de maneira gradual, sem depender de limiares rígidos (Mendel, 1995); sistemas de recomendação, nos quais as preferências dos usuários são consideradas de maneira aproximada (Bobadilla et al., 2013); agricultura de precisão, que exige a interpretação de variáveis como acidez e fertilidade do solo de forma não rígida (Shanthi e Gomathi, 2015); e diagnósticos médicos, em que sintomas não se apresentam de maneira categórica (Torres et al., 2006).

A Figura 1 ilustra como uma variável imprecisa pode ser representada por meio de suas funções de pertinência. O gráfico mostra como diferentes intervalos de temperatura pertencem parcialmente às categorias *Muito Frio*, *Frio*, *Agradável*, *Quente* e *Muito Quente*. É possível observar que uma temperatura entre 15°C e 25°C pode ser classificada como *Agradável*, mas também ter certo grau de pertinência nas categorias *Frio* ou *Quente*, dependendo do ponto específico.

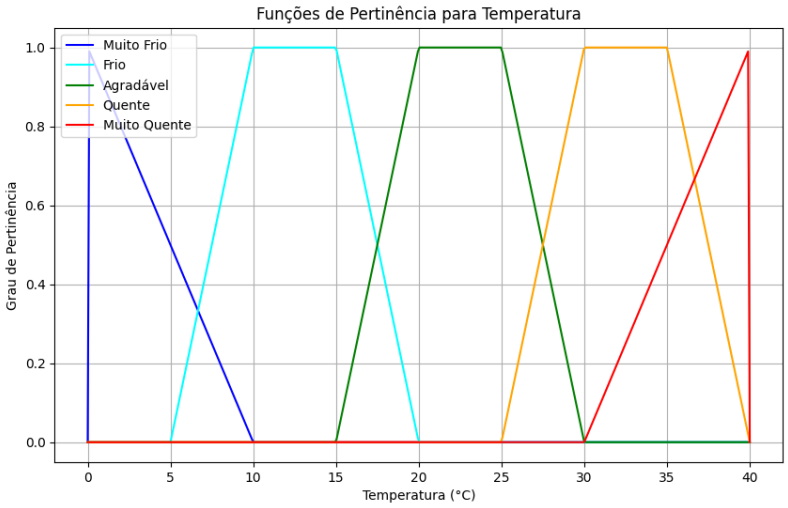


Figura 1. Funções de pertinência para temperatura (muito frio, frio, agradável, quente e muito quente). Fonte: Adaptado de Klir e Yuan (1995).

O processo de *fuzzyficação* é responsável por converter valores exatos em valores *fuzzy*, associando números precisos a categorias linguísticas com seus respectivos graus de pertinência. Por exemplo, considerando novamente a temperatura de um ambiente, percebe-se que 18°C possui aproximadamente grau 0.3 de *Frio* e 0.7 de *Agradável*. Esse tipo de categorização permite que o sistema tenha uma interpretação mais flexível dos dados de entrada, pois, após a *fuzzyficação*, os dados podem ser interpretados simultaneamente em múltiplas categorias *fuzzy*, com diferentes graus de pertinência.

No processo de inferência *fuzzy* é necessário combinar as informações *fuzzy* por meio de operadores lógicos. Os principais operadores são: *AND (E lógico), OR (OU lógico)* e *NOT (NÃO lógico)*. Segundo Klir e Yuan (1995), esses operadores podem ser implementados de diferentes formas, variando conforme o problema e o contexto. Para o operador *AND*, além do uso tradicional do mínimo, podem ser aplicadas outras abordagens, como o produto ou t-normas (normas triangulares) específicas. O operador *OR* pode ser implementado não apenas pelo máximo, mas também pela soma limitada ou outras t-normas. Já o *NOT*, embora comumente representado pelo complemento $(1-\mu_A(x))$, também admite funções de negação alternativas, que podem suavizar ou intensificar o efeito do complemento. As fórmulas dos operadores são implementadas da seguinte maneira:

- *AND* (*E lógico*):

$$\mu_{A\cap B}(x) = \begin{cases} \min(\mu_A(x), \mu_B(x)) & \text{(Mínimo)} \\ \mu_A(x) \cdot \mu_B(x) & \text{(Produto)} \end{cases} \tag{2}$$

- *OR* (*OU lógico*):

$$\mu_{A\cup B}(x) = \begin{cases} \max(\mu_A(x), \mu_B(x)) & \text{(Máximo)} \\ \min(1, \mu_A(x) + \mu_B(x)) & \text{(Soma Limitada)} \end{cases} \tag{3}$$

- *NOT* (*NÃO lógico*):

$$\mu_{\neg A}(x) = \begin{cases} 1 - \mu_A(x) & \text{(Complemento Padrão)} \\ (1 - \mu_A(x))^p & \text{(Parâmetro de Intensidade)} \end{cases} \tag{4}$$

Esses operadores permitem a formulação de regras complexas que integram múltiplas variáveis. A combinação entre *temperatura* e *umidade*, por meio do operador lógico *AND*, possibilita uma avaliação mais precisa do nível de conforto térmico, considerando a interação simultânea desses fatores. O Quadro 1 apresenta algumas regras criadas relacionando as variáveis temperatura e umidade com o operador lógico *E (AND)*, que levam à conclusão sobre a variável *conforto*.

A Figura 2 ilustra um exemplo de combinação *fuzzy* usando a operação *AND*. O gráfico representa a interação entre a temperatura e a umidade, mostrando o grau de pertinência resultante da combinação *fuzzy*. Áreas mais elevadas indicam combinações com maior associação segundo as regras estabelecidas.

Quadro 1. Regras *fuzzy* utilizando o operador lógico *E (AND)* entre temperatura e umidade, determinando o nível de conforto.

SE a temperatura for baixa E a umidade for baixa, ENTÃO o conforto é desconfortável.
SE a temperatura for baixa E a umidade for média, ENTÃO o conforto é neutro.
SE a temperatura for baixa E a umidade for alta, ENTÃO o conforto é desconfortável.
SE a temperatura for média E a umidade for baixa, ENTÃO o conforto é neutro.
SE a temperatura for média E a umidade for média, ENTÃO o conforto é confortável.
SE a temperatura for alta E a umidade for alta, ENTÃO o conforto é desconfortável.

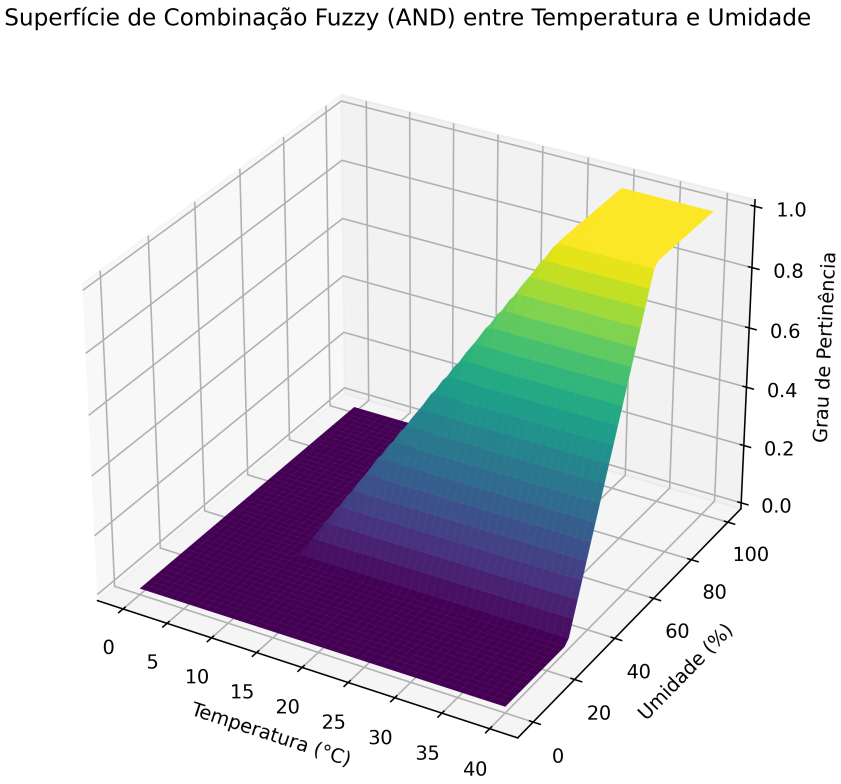


Figura 2. Superfície de combinação *fuzzy* (AND) entre temperatura e umidade. Fonte: Adaptado de Klir e Yuan (1995).

Após a inferência *fuzzy*, a saída do sistema continua sendo um conjunto de valores *fuzzy*, ou seja, é representada pelos graus de pertinência de todas as regras disparadas, distribuídos ao longo de uma faixa de possibilidades. Isso significa que, em vez de se obter um único valor definido como resposta, o sistema indica que a resposta pertence, com diferentes intensidades, a várias categorias.

No entanto, na maioria das aplicações práticas, é necessário obter uma ação ou decisão concreta, que não pode ser expressa em termos vagos. Por exemplo, em um sistema de controle de conforto térmico, após o processamento *fuzzy*, o sistema pode indicar que o ambiente está 0.6 *quente* e 0.4 *agradável* — entretanto, ele precisa converter essa informação em uma saída precisa, como a temperatura exata a ser ajustada no ar-

condicionado, resultando em um único valor contínuo, e não em graus de pertinência.

A *defuzzyficação* é o processo de converter os resultados *fuzzy*, representados por distribuições de pertinência, em um valor preciso e determinístico, o qual pode ser utilizado para a tomada de decisões em sistemas práticos. O método mais comum de *defuzzyficação* é o método do centróide (também chamado de centro de gravidade), que calcula a média ponderada da distribuição *fuzzy*. Esse método considera toda a área sob a curva das pertinências das regras disparadas, encontrando o ponto de equilíbrio da figura.

A Figura 3 ilustra o processo de *defuzzyficação*. Inicialmente, observa-se a distribuição *fuzzy* da variável de saída, na qual múltiplas categorias, como *frio*, *agradável* e *quente*, podem ser ativadas simultaneamente com diferentes graus de pertinência. A área sombreada representa essa sobreposição de categorias antes da *defuzzyficação*. Posteriormente, essa distribuição é convertida em um único valor *crisp*, indicado pela linha vertical na Figura 3, que corresponde ao resultado final utilizado para a tomada de decisão no sistema. Este valor é calculado como sendo o centro de gravidade da área cinza.

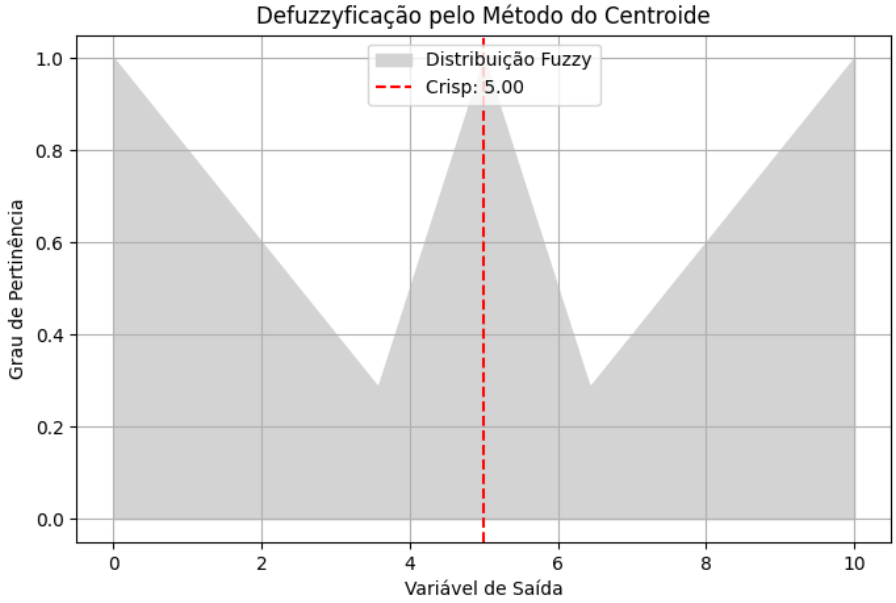


Figura 3. Comparação entre o estado *fuzzy* (área preenchida) e o valor *crisp* após *defuzzyficação*. Fonte: Adaptado de Ross (2010).

Em resumo, a *defuzzyficação* é essencial para converter as respostas difusas em resultados práticos e exatos, viabilizando a aplicação de sistemas *fuzzy* em situações do mundo real.

2.3. Trabalhos similares

Nesta seção, são apresentados diversos estudos que aplicaram sistemas baseados em lógica *fuzzy* no contexto agrícola, com o objetivo de otimizar práticas de manejo, avaliar atributos do solo e apoiar a tomada de decisão. As pesquisas revisadas demonstram como a lógica *fuzzy* tem sido eficaz na modelagem de cenários complexos e incertos, contribuindo para maior eficiência, sustentabilidade e precisão na agricultura.

Silva e Lima (2009) apresentam a aplicação da lógica *fuzzy* na avaliação da fertilidade do solo em uma área cultivada com café arábica variedade Catucaí. Foram coleta-

das amostras de solo em uma malha com 50 pontos sob a projeção da copa das plantas, analisando-se três atributos: Soma de Bases (SB), Capacidade de Troca de Cátions (CTC) e saturação por bases (V). Os dados passaram por análises descritivas e exploratórias, seguidas da integração dos valores dos atributos por meio de um sistema de classificação *fuzzy*. Posteriormente, realizou-se uma análise geoestatística para quantificar o grau de dependência espacial das possibilidades dos atributos.

Os resultados indicaram que a área estudada possui baixa possibilidade de desenvolvimento e rendimento da cultura, uma vez que os atributos avaliados apresentaram baixa disponibilidade. A utilização da lógica *fuzzy* permitiu representar a variabilidade espacial dos atributos do solo de forma mais precisa, oferecendo uma ferramenta eficaz para o mapeamento da fertilidade do solo e auxiliando na tomada de decisões agronômicas.

O estudo desenvolvido por Bönisch et al. (2004) apresenta uma metodologia para espacializar propriedades físico-químicas do solo expressas por atributos numéricos (teores de K e Al trocáveis, soma de bases, Capacidade de Troca Catiônica (CTC), saturação por bases e teor de areia total) e simultaneamente gerar mapas de incerteza espacial. Além disso, propõe modelar a propagação dessa incerteza por meio de procedimentos de álgebra de mapas baseados em lógica *fuzzy*, integrando-os em um sistema de informação geográfica.

O trabalho justifica-se pela necessidade de quantificar e representar a imprecisão inerente aos dados de solos em levantamentos pedológicos, superando abordagens qualitativas tradicionais. Ao incorporar medidas espaciais de incerteza e lógica *fuzzy*, busca-se fornecer um suporte analítico mais robusto para a tomada de decisão em zoneamento de fertilidade e manejo do solo, tratando de forma explícita as ambiguidades e variabilidades naturais dos atributos estudados.

Foram obtidos dados de 222 perfis pedológicos e 219 amostras extras em Santa Catarina. Cada atributo numérico foi transformado em conjuntos amostrais por indicação, definindo-se nove níveis de corte com igual número de amostras. Ajustaram-se semivariogramas teóricos (exponencial, gaussiano e esférico) aos semivariogramas experimentais e aplicou-se krigagem ordinária por indicação para estimar, em cada ponto, a função de distribuição acumulada condicionada (fdpac). A partir daí, calculou-se média e variância locais (intervalos de confiança de 95%) e implementou-se operadores de álgebra de mapas *fuzzy* (mapeamento linear e soma ponderada) com propagação de incertezas pelo método de Taylor de primeira ordem.

Os mapas gerados revelaram ampla variabilidade espacial dos atributos, com coeficientes de variação elevados indicando grande componente aleatória. As representações de incerteza (2σ) atingiram até 58% para o teor de areia total e 1096% de coeficientes de variação para a soma de bases, evidenciando a importância de incorporar esses indicadores. A integração *fuzzy* dos atributos produziu uma classificação contínua de limitação em fertilidade e mapeou a propagação das variâncias, demonstrando a viabilidade de apoiar decisões agronômicas com informação explícita sobre imprecisão.

No estudo de Santos et al. (2017), aplicam-se conceitos de geoestatística e geoprocessamento para delinear zonas de manejo em uma pastagem de capim Tanzânia em São Carlos – SP, definindo unidades de aplicação de corretivos (calagem) e fertilizantes (P e K). Para isso, ele compara métodos de interpolação espacial (krigagem ordinária e

IDW) e emprega lógica *fuzzy* na classificação e combinação dos mapas químicos do solo.

O trabalho visa otimizar o uso de insumos agrícolas em sistemas de produção intensiva a pasto, considerando a variabilidade espacial dos atributos do solo. Ao incorporar geoestatística e lógica *fuzzy*, busca-se fornecer suporte analítico robusto para decisões estratégicas de manejo, promovendo maior eficiência produtiva, redução de custos e minimização de impactos ambientais.

Os autores coletaram amostras de solo (seis subamostras / piquete) em um Latossolo Vermelho-Amarelo de pastagem irrigada, determinaram os atributos químicos (pH, MO, P, K, Ca, Mg, CTC, V) e ajustaram semivariogramas experimentais a modelos teóricos (esférico, gaussiano, exponencial). Cada modelo foi validado por cruzamento (erro médio absoluto, R^2, r e RQME). Em ArcGIS 10.2, cada mapa químico foi convertido em valores *fuzzy* (0–1) e combinado por soma ponderada, gerando cinco classes de fertilidade como zonas de manejo.

Foram obtidas cinco zonas de fertilidade: *muito baixa* (0,02 ha; 1,2%), *baixa* (0,3 ha; 18%), *média* (0,75 ha; 44%), *alta* (0,55 ha; 32%) e *muito alta* (0,08 ha; 4,8%). A krigagem ordinária apresentou melhor ajuste e valores de correlação forte ($r > 0,8$) frente aos mapas empíricos. A lógica *fuzzy* permitiu classificar de modo contínuo a limitação de fertilidade. Estimou-se, também, a economia de corretivos em aplicação variável (por ex., 7,84 t de calcário poupadas em 9,6 ha) e mapas de recomendação de adubação com alta confiabilidade, comprovando a viabilidade da geostatística e GIS *fuzzy* para decisões de agricultura de precisão.

Segundo Godinho et al. (2022), no contexto do agronegócio, o foco do estudo está na otimização da produtividade da cenoura. O trabalho propõe um sistema de regras *fuzzy* para definir os melhores parâmetros de manejo, especialmente o espaçamento entre plantas e a dosagem de adubo orgânico bovino. As variáveis de entrada — espaçamento (cm) e adubação (g/m²) — foram analisadas com base em dados experimentais de campo, e a variável de saída foi a produtividade da cenoura (ton/ha). O modelo usou o método de inferência de *Mamdani*, conhecido pela sua capacidade de lidar com incertezas e simular o raciocínio humano.

Neste trabalho, a lógica *fuzzy* foi empregada para construir um sistema de inferência baseado em regras do tipo *SE-ENTÃO*, com o objetivo de identificar as melhores combinações entre espaçamento das plantas e quantidade de adubo para maximizar a produtividade da cenoura. A modelagem considerou cinco faixas de espaçamento (de 2 a 14 cm) e cinco níveis de adubação (de 0 a 150 g/m²), todos representados por funções de pertinência triangulares. A produtividade, variável de saída, foi dividida em onze faixas (de 20 a 32 ton/ha). A inferência *fuzzy* possibilitou a geração de uma superfície de resposta e um mapa de contorno que indicaram claramente os pontos ideais de manejo.

Os resultados mostraram que os melhores níveis de produtividade ocorreram com espaçamentos entre 8,0 e 9,5 cm e adubação entre 100 e 130 g/m². A visualização tridimensional do modelo e o mapa de contorno facilitaram a identificação das combinações mais vantajosas, destacando a região com adubação alta e espaçamento médio como a mais promissora. A base de regras utilizada, composta por 25 regras *fuzzy*, foi implementada em ambiente *Matlab* e validada com dados experimentais de campo. O modelo apresentou bom desempenho e permitiu simular diferentes cenários de manejo, oferecendo

subsídios para práticas agrícolas mais eficientes.

O trabalho de Papadopoulos et al. (2011) apresenta o desenvolvimento de um sistema de apoio à decisão baseado em lógica *fuzzy* para recomendar doses de fertilização nitrogenada em culturas de algodão, considerando condições específicas de solo, clima e práticas agrícolas. O sistema utiliza uma estrutura em quatro níveis composta por onze subsistemas *fuzzy* que simulam o balanço de nitrogênio no solo, incorporando variáveis como mineralização, lixiviação, desnitrificação, volatilização e absorção pelas plantas. O modelo foi alimentado com 14 variáveis de entrada (agrupadas em categorias como características do solo, clima, irrigação e manejo de nutrientes) e construído com base em regras *fuzzy* obtidas por entrevistas com especialistas e literatura científica.

A avaliação do sistema foi realizada em 49 lavouras de algodão na região de Kopaida, Grécia, conhecida por seu risco de poluição por nitrato. Os resultados mostraram grande variação nas doses recomendadas (de 59,9 a 191,1 kg/ha), com uma média inferior à aplicada normalmente pelos agricultores da região. A análise de sensibilidade identificou que as variáveis mais influentes foram a precipitação média, temperatura do solo, textura e matéria orgânica. O sistema demonstrou potencial para reduzir a superdosagem de nitrogênio, propondo um manejo mais sustentável.

Em Choueri (2024), foram desenvolvidos dois sistemas baseados em regras *fuzzy* para apoiar a adubação nitrogenada da cultura do milho, um voltado à ureia e outro ao sulfato de amônio. A pesquisa justifica-se pela dificuldade de adaptar recomendações de doses de N às condições variáveis de solo e clima, pelo desperdício e pelas perdas de nitrogênio por volatilização e lixiviação, e pela necessidade de cumprir metas dos Objetivos de Desenvolvimento Sustentável (ODS 2 e 12), fornecendo uma ferramenta preditiva que antecipe cenários antes da aplicação em campo.

Foram construídos dois sistemas baseados em regras *fuzzy* com uma entrada (dose em 0, 40, 80, 120 e 160 Kg ha^{-1}) e 13 saídas (variáveis agronômicas como o número de espigas por planta, o índice de área foliar, diâmetro da espiga etc.), definindo cinco funções de pertinência gaussianas para a entrada e funções trapezoidais/triangulares para as saídas.

Os resultados mostraram que a ureia, principal fonte estudada, quando processada pelos sistemas baseados em regras *fuzzy*, prevê, por exemplo, diâmetro de colmo de até 25,8 cm com 120 Kg ha^{-1} (vs. 19,66 cm em Carvalho et al. (2019)), 670 grãos por espiga com 160 Kg ha^{-1} (vs. 561 GPE na literatura) e produtividade acima de 8.700 Kg ha^{-1}. O modelo para sulfato de amônio também possibilita simular cenários de dose *versus* resposta agronômica. Essas previsões permitem otimizar a dose antes da aplicação, reduzir perdas de N e contribuir para um manejo mais sustentável.

Dessa forma, observa-se que os estudos de Silva e Lima (2009), Bönisch et al. (2004) e Santos et al. (2017) estão voltados principalmente para aplicações em agricultura de precisão e elaboração de mapas de fertilidade. Por outro lado, os estudos de Godinho et al. (2022), Papadopoulos et al. (2011) e Choueri (2024) buscaram avaliar a correção do solo para determinada cultura. Entretanto, nenhum dos trabalhos revisados apresentou foco na análise crítica e avaliação quantitativa das tabelas presentes nos manuais oficiais de recomendação de adubação e calagem, que é justamente o objetivo central deste estudo.

3. Desenvolvimento do sistema Arvo

O sistema de recomendação de adubação Arvo foi estruturado em duas camadas principais: *fron-tend* e *back-end*. O *back-end*, desenvolvido em *Python* com *FastAPI*, é responsável pelo processamento dos dados, aplicação das regras *fuzzy* e definição das recomendações de adubação. O *front-end*, desenvolvido com *Vue3* e *TypeScript*, permite a interação do usuário com o sistema, incluindo o *upload* de laudos em formato PDF e a visualização das recomendações geradas.

Esta seção apresenta o desenvolvimento do sistema de recomendação de adubação baseado em lógica *fuzzy*. Inicialmente, são definidos os requisitos funcionais e não funcionais que guiaram o processo de construção da solução. Em seguida, é detalhada a arquitetura do sistema, além das tecnologias adotadas para sua implementação. Também são apresentados o modelo relacional, que representa a estrutura lógica da aplicação, e os principais aspectos da implementação.

3.1. Definição dos requisitos

A definição dos requisitos foi uma etapa essencial para orientar o desenvolvimento do sistema de recomendação de adubação com base em lógica *fuzzy*. Esses requisitos demonstram as características necessárias para que o sistema atenda adequadamente às demandas técnicas e funcionais, além de assegurar sua qualidade, desempenho e segurança. Os requisitos foram obtidos em reuniões com um engenheiro agrônomo especialista em correção de solo, que contribuiu com informações práticas, baseadas em sua experiência, além de esclarecer dúvidas técnicas durante o processo de desenvolvimento. Em paralelo, foi utilizado como fonte o manual da Sociedade Brasileira de Ciência do Solo (2016), que serviu de referência para consolidar e validar os critérios adotados na implementação do sistema.

Os requisitos foram organizados em duas categorias principais: funcionais, que descrevem o conjunto de funcionalidades necessárias para que o sistema cumpra seus objetivos; e não funcionais, que definem os atributos de qualidade que devem ser observados durante o desenvolvimento e a implantação. A definição contemplou aspectos como a extração automática de dados a partir de relatórios de análise de solo, a aplicação do modelo de inferência *fuzzy*, a apresentação das recomendações ao usuário, bem como requisitos de segurança, desempenho e usabilidade.

Todos os requisitos identificados no processo de especificação estão organizados nos Quadros 2 e 3, que sintetizam as funcionalidades e qualidades essenciais que nortearam o desenvolvimento do sistema. A partir da definição dos requisitos, assegurou-se que o sistema fosse capaz de atender às demandas específicas do setor agrícola, mantendo flexibilidade e robustez para suportar futuras evoluções.

3.2. Arquitetura do sistema

A arquitetura do sistema foi projetada no modelo cliente-servidor, com duas camadas: *front-end* e *back-end*. Essa separação organiza o sistema e facilita sua manutenção e expansão.

Quadro 2. Requisitos funcionais do sistema de recomendação de adubação.

Nome	Descrição
RF 1 – Gerenciar Produtor	O sistema deve permitir o cadastro, visualização, edição e remoção (CRUD) da entidade *Produtor*, com os seguintes campos: nome, CPF, telefone, *e-mail*, *status* e vínculo com o agrônomo responsável.
RF 2 – Gerenciar Propriedade	O sistema deve permitir o cadastro, visualização, edição e remoção (CRUD) da entidade *Propriedade*, com os seguintes campos: nome, endereço, cidade, área (hectares), latitude, longitude e vínculo com o produtor.
RF 3 – Gerenciar Laboratório	O sistema deve permitir o cadastro, visualização, edição e remoção (CRUD) da entidade *Laboratório*, com os seguintes campos: nome, endereço, cidade, telefone, *e-mail*, campo *is_global* e vínculo com o agrônomo.
RF 4 – Gerenciar Cultivo	O sistema deve permitir o cadastro, visualização, edição e remoção (CRUD) da entidade *Cultivo*, com os seguintes campos: nome do cultivo, percentuais desejados de N, P, K e pH ideal.
RF 5 – Gerenciar Usuário	O sistema deve permitir o cadastro, visualização, edição e remoção (CRUD) da entidade *Usuário* e seu *Perfil*, incluindo nome, CPF, telefone, *e-mail* e logotipo no sistema.
RF 6 – Importar Laudos PDF	O sistema deve permitir que o agrônomo faça o *upload* de laudos em PDF, com extração automática das informações técnicas quando possível.
RF 7 – Cadastrar Manualmente Laudos	O sistema deve permitir o cadastro manual de laudos (*Laudo*) e suas respectivas amostras (*Amostra*) para casos em que o *upload* não seja viável.
RF 8 – Gerar Recomendações	O sistema deve permitir gerar recomendações de adubação e calagem vinculadas à entidade *Recomendação*, com base nos dados da *Amostra* e nos parâmetros do *Cultivo*.
RF 9 – Calcular Adubação pelo Manual	O sistema deve realizar cálculos de adubação com base no Manual de Calagem e Adubação para os estados de SC e RS. Os resultados devem ser armazenados na entidade *Recomendação*.
RF 10 – Calcular Adubação via *Fuzzy*	O sistema deve realizar cálculos de adubação utilizando inferência baseada em lógica *fuzzy* aplicada às tabelas do manual. Os resultados devem ser armazenados na entidade *Recomendação*.
RF 11 – Calcular Calagem	O sistema deve realizar cálculos de calagem usando os métodos Saturação por Bases e SMP, vinculados à entidade *Recomendação*.

Quadro 3. Requisitos não funcionais do sistema de recomendação de adubação.

Nome	Descrição
RNF 1 – Arquitetura em Camadas	O sistema deve ser estruturado em duas camadas principais: apresentação *front-end* com *Vue3* e *TypeScript* e processamento *back-end* em *Python* com *FastAPI*. A comunicação entre as camadas ocorre via API REST.
RNF 2 – Segurança de Dados	O acesso deve ser protegido com autenticação JWT, comunicação via HTTPS (TLS 1.3), criptografia *AES-256* e senhas armazenadas com *bcrypt*.
RNF 3 – Integridade e Consistência	O sistema deve realizar validação automática dos dados inseridos nas entidades (*Laudo, Amostra, Propriedade*) para garantir a integridade e a consistência das informações.
RNF 4 – Interface Responsiva	A interface deve ser responsiva e adaptável a diferentes dispositivos e tamanhos de tela.
RNF 5 – Controle de Acesso	Apenas usuários autenticados devem ter acesso às funções protegidas, mediante autenticação JWT.
RNF 6 – Isolamento de Dados	Cada agrônomo deve ter acesso exclusivo aos dados vinculados à sua conta.
RNF 7 – Implantação em Nuvem	O sistema deve ser hospedado em ambiente de nuvem (*AWS* ou similar), garantindo escalabilidade e disponibilidade.

A comunicação entre *front-end* e *back-end* ocorre via requisições HTTPS utilizando uma API REST, que envia e recebe dados no formato JSON. O *front-end* foi desenvolvido com *Vue3* e *TypeScript*, para possibilitar que o usuário insira os dados da análise de solo e visualize as recomendações. O *back-end* é uma API REST desenvolvida em *Python* com *FastAPI*. Nele estão implementadas as regras de negócio, os cálculos agronômicos e os modelos de inferência *fuzzy*, utilizando a biblioteca *scikit-fuzzy*[2]. O *back-end* processa os dados recebidos, aplica os métodos tradicionais ou o modelo *fuzzy* para o cálculo de recomendações e retorna os resultados. O Sistema Gerenciador de Banco de Dados (SGBD) utilizado é *PostgreSQL*, onde ficam armazenados os dados dos usuários, análises de solo e histórico de inferências.

O fluxo do sistema funciona da seguinte forma: o usuário insere os dados da análise de solo no *front-end*. O *front-end* envia esses dados para o *back-end* via requisições HTTPS. O *back-end* processa os dados, acessa o banco de dados quando necessário e executa os cálculos para gerar as recomendações. O *back-end* responde ao *front-end* com os resultados em JSON e o *front-end* exibe ao usuário as recomendações geradas. A Figura 4 ilustra a arquitetura do sistema. Essa arquitetura facilita sua manutenção, organização e expansão futura.

[2]Documentação oficial da biblioteca: `https://scikit-fuzzy.readthedocs.io/en/latest/`

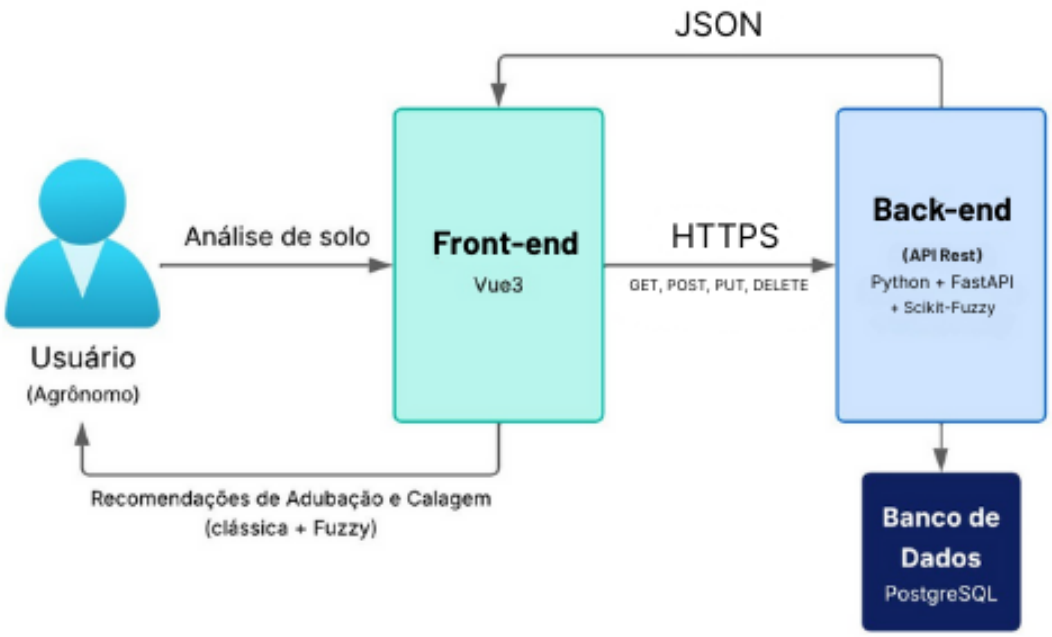


Figura 4. Arquitetura do sistema.

3.3. Tecnologias utilizadas

A escolha das ferramentas para o desenvolvimento do sistema de recomendação de adubação levou em consideração não apenas critérios técnicos, mas também a aderência às necessidades específicas do projeto, especialmente no que se refere à construção da lógica *fuzzy* e ao desenvolvimento de uma interface web funcional e acessível.

Na camada de *back-end*, optou-se pela linguagem de programação *Python* e *FastAPI*, que possuem uma vasta gama de bibliotecas voltadas para ciência de dados, inteligência artificial e desenvolvimento de APIs. No contexto deste sistema, a utilização da biblioteca *scikit-fuzzy (skfuzzy)* foi essencial, pois ela oferece ferramentas para a modelagem de sistemas baseados em lógica *fuzzy*, que são a base do motor de recomendação. Através dela, foi possível definir as variáveis linguísticas, construir as funções de pertinência, estabelecer as regras *fuzzy* e realizar a inferência, permitindo transformar dados quantitativos de análise de solo em recomendações qualitativas e personalizadas de adubação.

Na camada de *front-end*, o sistema foi construído utilizando o *framework Vue.js (Vue3)*, que se baseia na criação de componentes reativos e modulares. Essa escolha proporciona uma interface dinâmica e responsiva. Além disso, foi incorporado o uso de *TypeScript*, uma extensão do *JavaScript* que adiciona tipagem estática ao código. Isso trouxe maior segurança e confiabilidade durante o desenvolvimento, reduzindo a incidência de erros, facilitando a detecção de inconsistências e melhorando a organização do código.

As tecnologias HTML e CSS foram responsáveis pela estruturação e estilização da interface web. O uso de boas práticas de CSS, juntamente com *frameworks* de *layout* responsivo, garantiu que a aplicação se adapte corretamente a diferentes tamanhos de tela, oferecendo uma experiência fluida tanto em dispositivos *desktop* quanto móveis.

Além disso, foi adotada a ferramenta de conteinerização *Docker*, que permite encapsular a aplicação juntamente com todas as suas dependências, bibliotecas e configurações. Isso garante que o sistema funcione de forma idêntica em qualquer ambiente, seja local, de teste ou de produção. Essa abordagem também facilita a escalabilidade do sistema, além de simplificar a manutenção e o processo de *deploy*.

3.4. Modelo relacional

A Figura 5 apresenta o modelo relacional do sistema desenvolvido; alguns atributos das tabelas *Amostra* e *Recomendacao* foram omitidos para melhorar a legibilidade da figura. O usuário (tabela *User*) representa o ponto de entrada no sistema, armazenando informações básicas de autenticação. Ele está associado ao perfil (tabela *Perfil*), que concentra dados pessoais e de identificação, como nome, CPF, telefone e logotipo, contemplando o RF 5 – Gerenciar Usuário.

O produtor (tabela *Produtor*) representa o cliente agrícola, com atributos como nome, CPF, telefone, *e-mail* e *status*, além de estar vinculado ao agrônomo responsável. Esse relacionamento assegura o atendimento ao RF 1 – Gerenciar Produtor. Cada produtor pode estar associado a uma ou mais propriedades, a tabela *Propriedade* registra dados como nome, endereço, área, cidade, latitude e longitude, atendendo ao RF 2 – Gerenciar Propriedade e permitindo a correta identificação espacial das áreas de cultivo.

A propriedade (tabela *Propriedade*) se relaciona diretamente com o laudo (tabela *Laudo*), que são documentos de análise de solo que podem ser inseridos manualmente ou importados de arquivos PDF, correspondendo aos RF 6 – Importar Laudos PDF e RF 7 – Cadastrar Manualmente Laudos, além disso, a entidade *LaudoPdf* é responsável por armazenar os arquivos originais dos laudos em formato digital, garantindo a integridade e o acesso posterior aos documentos importados. Os laudos estão vinculados à entidade *Laboratorio*, responsável por centralizar as análises realizadas. Essa entidade mantém informações como nome, endereço, contatos e abrangência de atuação, garantindo o RF 3 – Gerenciar Laboratório. Cada laudo contém uma ou mais amostras, a entidade *Amostra* armazena os valores de nutrientes e propriedades químicas do solo (pH, Ca, Mg, K, matéria orgânica, entre outros). Essas amostras se relacionam a um cultivo (tabela *Cultivo*), que define a cultura agrícola e seus parâmetros ideais de nutrientes e pH, atendendo ao RF 4 – Gerenciar Cultivo.

Com base nos laudos, amostras e cultivos cadastrados, o sistema gera uma recomendação (tabela *Recomendacao*), consolidando os cálculos de adubação e calagem. Essa entidade está diretamente vinculada aos RF 8 – Gerar Recomendações, RF 9 – Calcular Adubação pelo Manual, RF 10 – Calcular Adubação via *fuzzy* e RF 11 – Calcular Calagem, armazenando resultados detalhados, como necessidades de nutrientes, doses de fertilizantes e corretivos por hectare e totais.

Por fim, as entidades *Estado* e *Cidade* oferecem suporte à organização geográfica, vinculando-se a propriedade (tabela *Propriedade*) e laboratório (tabela *Laboratorio*), assegurando consistência e integridade nos cadastros. Assim, o modelo relacional estabelece a base estrutural necessária para a implementação do sistema, garantindo que cada requisito funcional seja contemplado desde o gerenciamento das entidades até a geração automatizada de recomendações.

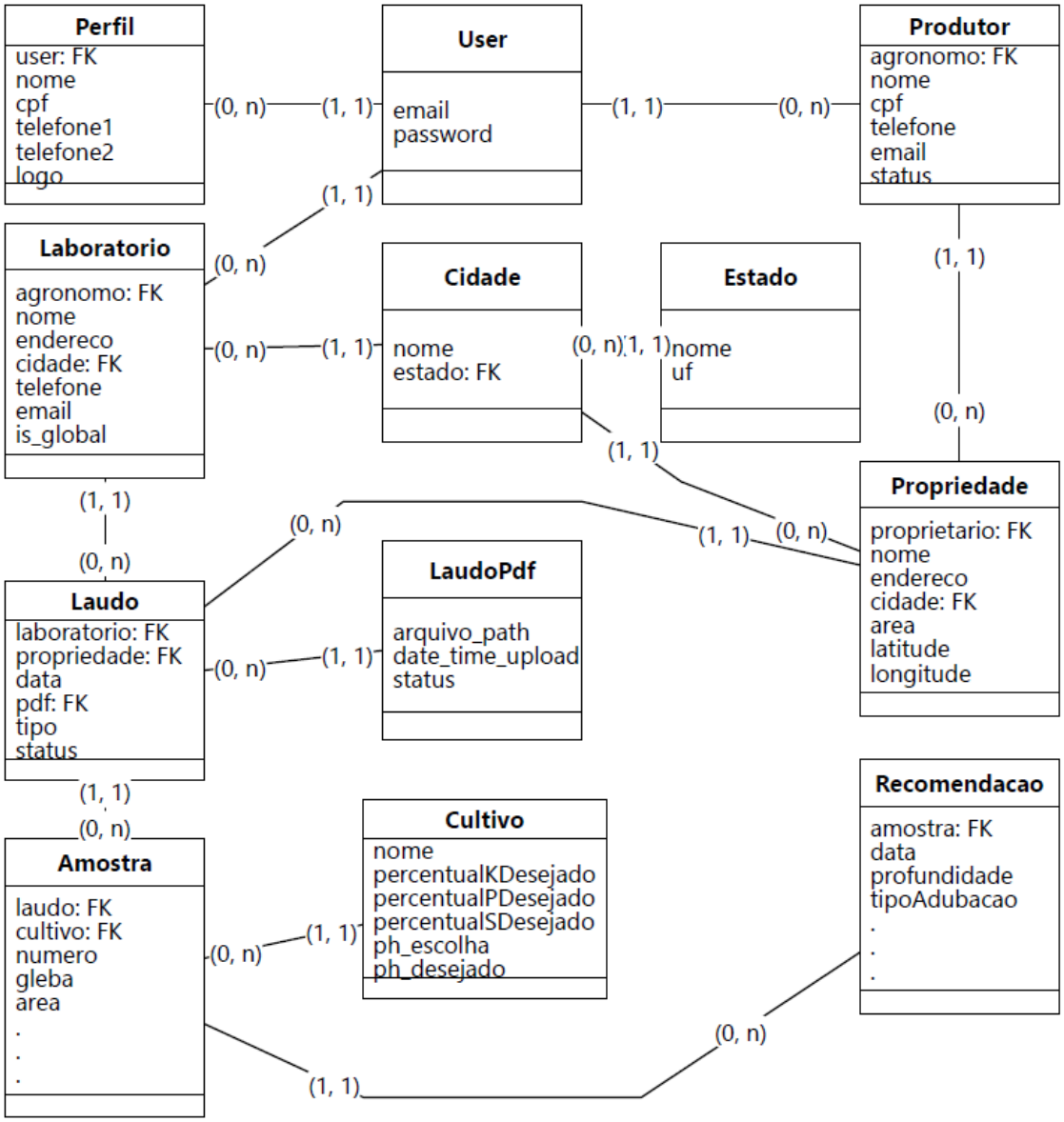


Figura 5. Modelo relacional do sistema.

3.5. Implementação

De maneira geral, os cadastros do sistema seguem uma estrutura semelhante, envolvendo interface web, chamadas ao *back-end* e persistência no banco de dados. Para exemplificar esse funcionamento, esta seção apresenta o cadastro de propriedade (tabela *Propriedade*). O formulário permite inserir dados como nome, endereço, telefone, *e-mail*, cidade, estado, área e vincular o produtor (tabela *Produtor*), atendendo à chave estrangeira que mantém a integridade do relacionamento.

O *front-end* gerencia o estado da aplicação, incluindo a lista de propriedades e a abertura dos modais para cadastro, edição e consulta. As funções *cadastrarPropriedade* e *editarPropriedade*, nas linhas 6 e 11, respectivamente, do Quadro 4, realizam chamadas às funções do *back-end*, garantindo que, após a operação, a lista seja atualizada auto-

maticamente e o usuário receba *feedback* visual da operação por meio de notificações *toasts*. As funções *acessarEdicaoPropriedade*, *acessarCadastroPropriedade* e *acessarConsultaPropriedade*, nas linhas 23, 26 e 29, respectivamente, do Quadro 4, controlam a visibilidade dos modais e identificam a propriedade a ser manipulada.

Quadro 4. Código completo do *composable* de *Propriedades*.

```
1    [...]
2    const { getPropriedadeList, deletePropriedade,
     ↪ postPropriedade, putPropriedade } =
     ↪ usePropriedadeRequisicoes();
3    const getPropriedades = async () => {
4      const response = await getPropriedadeList();
5      propriedades.value = response; };
6    const cadastrarPropriedade = async (propriedade:
     ↪ PostPropriedadeInterface) => {
7      await postPropriedade(propriedade);
8      modalCadastroOpen.value = false;
9      $toast.success('Propriedade Cadastrada com sucesso');
10     await getPropriedades(); };
11   const editarPropriedade = async (propriedade:
     ↪ PostPropriedadeInterface) => {
12     if(!idPropriedadeEdicao.value){
13       $toast.error('Falha ao salvar alterações, tente
     ↪ novamente mais tarde');
14       return; }
15     await putPropriedade(idPropriedadeEdicao.value,
     ↪ propriedade);
16     modalCadastroOpen.value = false;
17     $toast.success('Propriedade Atualizada com sucesso');
18     await getPropriedades(); };
19   const excluirPropriedade = async (propriedadeId: number) =>
     ↪ {
20     await deletePropriedade(propriedadeId);
21     $toast.success('Propriedade excluída com sucesso');
22     await getPropriedades(); };
23   const acessarEdicaoPropriedade = (propriedadeId: number) =>
     ↪ {
24     idPropriedadeEdicao.value = propriedadeId;
25     modalCadastroOpen.value = true; };
26   const acessarCadastroPropriedade = () => {
27     idPropriedadeEdicao.value = undefined;
28     modalCadastroOpen.value = true; };
29   const acessarConsultaPropriedade = (propriedadeId: number)
     ↪ => {
30     modalConsultaOpen.value = true;
31     idPropriedadeConsulta.value = propriedadeId; };
32
```

O Quadro 5 exemplifica como as chamadas do *front-end* são direcionadas para funções específicas do *back-end*, que realizam a persistência e recuperação de dados no banco. As operações de criação, edição e consulta detalhada de uma propriedade são implementadas na camada de acesso ao banco, mantendo os relacionamentos com *Produtor, Cidade* e *Estado*, garantindo a consistência referencial. Por exemplo, na função *create_propriedade* (linha 1 do Quadro 5), o objeto *Propriedade* é instanciado com os dados recebidos do *front-end*. Em seguida, a propriedade é adicionada à sessão e persistida no banco por meio dos comandos *session.add(propriedade)* e *await session.commit()*, retornando posteriormente o ID gerado, que confirma a criação bem-sucedida da entidade.

Quadro 5. Função assíncrona para criar uma propriedade no banco de dados.

```
1     async def create_propriedade(
2         propriedade_payload: PropriedadeCreate,
3         session: AsyncSession,
4     ) -> ResponsePostUpdateDelete:
5         propriedade = Propriedade(
6             produtor_id=propriedade_payload.produtor_id,
7             nome=propriedade_payload.nome,
8             endereco=propriedade_payload.endereco,
9             cidade_id=propriedade_payload.cidade_id,
10            area=propriedade_payload.area,
11            latitude=propriedade_payload.latitude,
12            longitude=propriedade_payload.longitude, )
13        session.add(propriedade)
14        await session.commit()
15        await session.refresh(propriedade)
16        if propriedade.id is None:
17            raise ValueError("Falha ao criar propriedade - ID
  ↪ não foi gerado")
18        return ResponsePostUpdateDelete(id=propriedade.id)
19
```

As rotas definidas no Quadro 6 expõem os *endpoints* da API que permitem realizar operações de CRUD e consultas detalhadas sobre propriedades. Quando o *front-end* realiza uma chamada, como cadastrar uma nova propriedade, o pedido é recebido pelo *endpoint* correspondente no *back-end*. Este *endpoint* é responsável por validar a autenticação do usuário, iniciar a sessão do banco de dados e invocar a função de negócio que realiza a operação desejada.

No caso do cadastro de uma propriedade, a função *create_propriedade* recebe os dados enviados pelo *fron-tend*, instancia o objeto *Propriedade*, valida os relacionamentos com *Produtor*, *Cidade* e *Estado*, e persiste as informações no banco. Uma vez que a operação é concluída, o *endpoint* retorna uma resposta padronizada utilizando o *ResponseModel*, que contém informações como *status* da operação, mensagem descritiva e o ID gerado da propriedade. De maneira análoga, *endpoints* de atualização e consulta seguem o mesmo padrão: recebem os dados ou parâmetros do *front-end*, executam a lógica de negócios no *back-end*, e retornam uma resposta consistente e padronizada para o *front-end*, garantindo segurança, consistência referencial e uniformidade na comunicação.

Quadro 6. Rotas da API para *Propriedades*.

```
        response_model=ResponseModel[ResponsePostUpdateDelete],
        status_code=status.HTTP_201_CREATED,
        summary="Cadastrar propriedade", )
    async def post_create_propriedade(
        propriedade_payload: PropriedadeCreate,
        session: AsyncSession = Depends(get_session),
        current_user: UserRead = Depends(get_current_user), ):
        created = await create_propriedade(
            propriedade_payload=propriedade_payload,
            session=session, )
        return ResponseModel(ok=True, message="Propriedade
    ↪ cadastrada", data=created)
    async def put_update_propriedade(
        propriedade_id: int,
        propriedade_payload: PropriedadeUpdate,
        session: AsyncSession = Depends(get_session),
        current_user: UserRead = Depends(get_current_user), ):
        success = await update_propriedade(
            propriedade_id=propriedade_id,
            propriedade_payload=propriedade_payload,
            user_logado_id=current_user.id,
            session=session, )
        if not success:
            raise HTTPException(
                status_code=status.HTTP_404_NOT_FOUND,
                detail="Propriedade não encontrada ou sem permissão", )
        return ResponseModel(
            ok=True,
            message="Propriedade atualizada",
            data=ResponsePostUpdateDelete(id=propriedade_id),
    [...]
    @router.get(
        "/{propriedade_id}/relatorio-consulta/",
        response_model=ResponseModel[PropriedadeRelatorioConsulta],
        summary="Buscar relatório de consulta de Propriedade", )
    async def get_relatorio_consulta_propriedade(
        propriedade_id: int,
        session: AsyncSession = Depends(get_session),
        current_user: UserRead = Depends(get_current_user), ):
        prop = await relatorio_consulta_propriedade(
            propriedade_id=propriedade_id,
            user_logado_id=current_user.id,
            session=session, )
        if not prop:
            raise HTTPException(
                status_code=status.HTTP_404_NOT_FOUND,
                detail="Propriedade não encontrada", )
        return ResponseModel(ok=True, message="Propriedade
    ↪ encontrada", data=prop)

```

O *Model Propriedade* representa a entidade principal no banco de dados, conforme ilustrado no Quadro 7. Ele herda de *PropriedadeBase*, que define os campos essenciais da tabela, como *nome, endereco, produtor_id, cidade _ID, area, latitude* e *longitude*. Dessa forma, a classe *Propriedade* combina esses atributos com o identificador único *id* e os relacionamentos com *Produtor* e *Laudo*, permitindo associar cada propriedade ao produtor correspondente e a múltiplos laudos. Essa estrutura garante que operações de cadastro, edição e consulta detalhada mantenham a consistência referencial e facilitem o carregamento de informações relacionadas.

Quadro 7. Definição do modelo ORM para a tabela de *Propriedade*.

```
    [...]
    class PropriedadeBase(SQLModel):
    nome: str = Field(max_length=255)
    endereco: str = Field(max_length=255)
    produtor_id: int = Field(foreign_key='produtores.id',
    ↪ ondelete='CASCADE')
    cidade_id: int = Field(foreign_key='cidades.id',
    ↪ ondelete='CASCADE')
    area: Optional[float] = None
    latitude: Optional[float] = None
    longitude: Optional[float] = None
    class Propriedade(PropriedadeBase, table=True):
        __tablename__: ClassVar[str] = 'propriedades'
        id: Optional[int] = Field(default=None, primary_key=True)
        produtor: 'Produtor' =
    ↪ Relationship(back_populates='propriedades')
        laudos: List['Laudo'] =
    ↪ Relationship(back_populates='propriedade')
    [...]

```

A importação dos laudos em formato PDF foi realizada utilizando a biblioteca *pdfplumber*, que permite extrair o conteúdo textual das páginas de maneira estruturada. O PDF é carregado em memória como *bytes*, evitando a criação de arquivos temporários, e aberto com o *pdfplumber* para leitura de cada página. O texto extraído é processado linha a linha, permitindo identificar informações como a data de emissão do laudo e os valores das amostras de solo. Durante o processamento, os valores numéricos são convertidos de *strings* para *float*, substituindo vírgulas por pontos, e valores ausentes são tratados como *None*, garantindo a integridade dos dados para análises posteriores.

3.6. Implementação dos cálculos de recomendação por meio das tabelas do MACSCRS

A implementação das recomendações baseadas nas tabelas do MACSCRS (Sociedade Brasileira de Ciência do Solo, 2016) foi realizada por meio de uma representação algorítmica das tabelas publicadas no manual. As estruturas implementadas encapsulam os intervalos e as classificações presentes nas tabelas do manual e permitem a consulta. Nesta seção descreve-se a implementação das tabelas e como elas são utilizadas.

Inicialmente, foi criada a classe *Tabela* para representar cada tabela do MACSCRS. Uma *Tabela* contém uma lista de *Linha*, cada *Linha* possui três atributos: *intervalo_min* (limite inferior), *intervalo_max* (limite superior — pode ser *None* para infinito)

e *classificacao* (rótulo da faixa). O Quadro 8 demonstra duas tabelas instanciadas como exemplo: a tabela de argila e uma tabela de disponibilidade de fósforo. A tabela de argila corresponde à coluna *Argila* da Tabela 2). Já a tabela de disponibilidade de fósforo corresponde à faixa de argila 3 da Tabela 3, utilizada no exemplo da seção 2.1.2.

Quadro 8. Tabelas representando as tabelas do MACSCRS.

```
    [...]
    self.tabela_argila = Tabela(
        id=1,
        nome="Argila: Tab. 6.1",
        unidade="%",
        linhas=[
            Linha(intervalo_min=0, intervalo_max=20,
↪ classificacao="4"),
            Linha(intervalo_min=21, intervalo_max=40,
↪ classificacao="3"),
            Linha(intervalo_min=41, intervalo_max=60,
↪ classificacao="2"),
            Linha(
                intervalo_min=60.1, intervalo_max=None,
↪ classificacao="1"), ],  )
    self.tabela_fosforo_grupo_1_classe_3 = Tabela(
        id=4,
        nome="Fósforo: Tab. 6.3 - Grupo 1 - Classe 3",
        unidade="mg/dm^3",
        linhas=[
            Linha(intervalo_min=0, intervalo_max=10,
↪ classificacao="muito_baixo"),
            Linha(intervalo_min=10.1, intervalo_max=20,
↪ classificacao="baixo"),
            Linha(intervalo_min=20.1, intervalo_max=30,
↪ classificacao="medio"),
            Linha(intervalo_min=30.1, intervalo_max=60,
↪ classificacao="alto"),
            Linha(
                intervalo_min=60.1, intervalo_max=None,
↪ classificacao="muito_alto"), ], )
    [...]

```

O código apresentado no Quadro 9 mostra a função responsável por calcular a recomendação de P_2O_5 a partir das entradas fornecidas. Em termos funcionais, a rotina realiza os seguintes passos:

1. Inicializa o repositório de tabelas de fósforo;
2. Busca a tabela de argila que é fixa e obtém a classe de argila do solo, passando como entrada a porcentagem de argila presente no solo;
3. Com essa classe de argila e o grupo de P do cultivo, previamente cadastrado, busca a tabela de disponibilidade de fósforo correspondente;
4. Classifica o teor de fósforo na tabela de disponibilidade, passando como entrada o fósforo presente no solo;
5. Obtém a tabela de saída específica para o cultivo e encontra a necessidade de P_2O_5 associada à classe de disponibilidade obtida.

Dessa forma, a função encadeia consultas às tabelas e cada chamada seleciona a tabela adequada com base nas variáveis de entrada e extrai o intervalo/valor final que compõe a recomendação.

Quadro 9. Tabelas representando as tabelas do MACSCRS.

```
    [...]
    def calcula_p2o5_recomendado(
        entrada_p2o5: EntradaP2O5RecomendadoManualScRs,
    ) -> SaidaP2O5Recomendado:
        tabelas_fosforo = TabelasFosforo()
        tabela_argila = tabelas_fosforo.tabela_argila
        classe_argila =
    ↪ tabela_argila.classificar_valor(round(entrada_p2o5.argila, 2))
        tabela_disponibilidade_escohida = (

    ↪ tabelas_fosforo.obter_tabela_fosforo_classe_disponibilidade(
                grupo=entrada_p2o5.grupo_p, classe=classe_argila)
        )
        classe_disponibilidade_p =
    ↪ tabela_disponibilidade_escohida.classificar_valor(
            round(entrada_p2o5.p, 2)
        )
        tabela_saida_escolhida =
    ↪ tabelas_fosforo.obter_tabela_fosforo_saida_p2o5(
            entrada_p2o5.cultivo_nome
        )
        necessidade_p2o5 =
    ↪ tabela_saida_escolhida.retorna_intervalo_min(
            classe_disponibilidade_p
        )
    [...]

```

Vale ressaltar que todas as operações de cálculos para o fósforo, exibidas nesta seção, são semelhantes às operações de potássio implementadas.

3.7. Implementação dos cálculos de recomendação por meio das tabelas *fuzzyficadas*

A implementação da lógica *fuzzy* no sistema foi feita por meio da biblioteca *scikit-fuzzy (skfuzzy)*. Nessa seção são abordadas as três principais etapas realizadas: (1) construção do sistema *fuzzy*; (2) montagem dos conjuntos *fuzzy* para cada *Antecedent* e *Consequent*; (3) *defuzzificação* e obtenção do valor numérico da recomendação.

O sistema *fuzzy* é construído pela função *build_sistema_fuzzy* apresentada no Quadro 10. O processo inicia-se com a definição dos universos de todas as variáveis usadas no sistema (linhas 6 a 8). Em seguida, são criados os *Antecedents* (variáveis de entrada no *fuzzy*) e *Consequents* (variáveis de saída no *fuzzy*), como pode ser visto nas linhas 9 a 24, e o conjunto de regras, armazenados na lista *rules* (linhas 25 a 36). Por fim, nas linhas 37 e 38 são efetivamente criados o sistema *fuzzy* e uma instância de simulação que é usada para processar os valores de entrada.

Quadro 10. Sistema *fuzzy* divido em duas funções.

```
1     [...]
2     def build_sistema_fuzzy(
3         tabela_classica_entrada: list[Tabela],
4         tabela_classica_saida: Tabela,
5     ) -> SistemaFuzzy:
6         x_argila = np.arange(4, 100, 1)
7         x_teor_fosforo = np.arange(0, 73, 0.1)
8         x_fosforo_por_ha = np.arange(-30, 280, 1)
9         argila = ctrl.Antecedent(x_argila, "teor_argila")
10        argila["faixa_4"] = fuzz.zmf(x_argila, 15, 25)
11        argila["faixa_3"] = fuzz.trapmf(x_argila, [17, 26, 36, 45])
12        argila["faixa_2"] = fuzz.trapmf(x_argila, [37, 46, 56, 62])
13        argila["faixa_1"] = fuzz.smf(x_argila, 57, 75)
14        teor_potassio_faixa_argila_1 = fuzzyfica_antecedent([...])
15        teor_potassio_faixa_argila_2 = fuzzyfica_antecedent([...])
16        teor_potassio_faixa_argila_3 = fuzzyfica_antecedent(
17            tabela_classica=tabela_classica_entrada[2],
18            universo_fuzzy=x_teor_fosforo,
19            nome_antecedent="teor_fosforo_faixa_argila_3",
20            list_classificacoes=["muito_baixo", "baixo", "medio",
    ↪ "alto", "muito_alto"],
21            porcentagem_plato=0.2,
22        )
23        teor_potassio_faixa_argila_4 = fuzzyfica_antecedent([...])
24        fosforo_por_ha = fuzzyfica_consequent([...])
25        rules = [
26            [...]
27            ctrl.Rule(
28                argila["faixa_3"] &
    ↪ teor_potassio_faixa_argila_3["muito_baixo"],
29                fosforo_por_ha["muito_alto"],
30            ),
31            ctrl.Rule(
32                argila["faixa_3"] &
    ↪ teor_potassio_faixa_argila_3["baixo"],
33                fosforo_por_ha["alto"],
34            ),
35            [...]
36        ]
37        fosforo_ctrl = ctrl.ControlSystem(rules)
38        fosforo_sim = ctrl.ControlSystemSimulation(fosforo_ctrl)
39        [...]
40    [...]
41
```

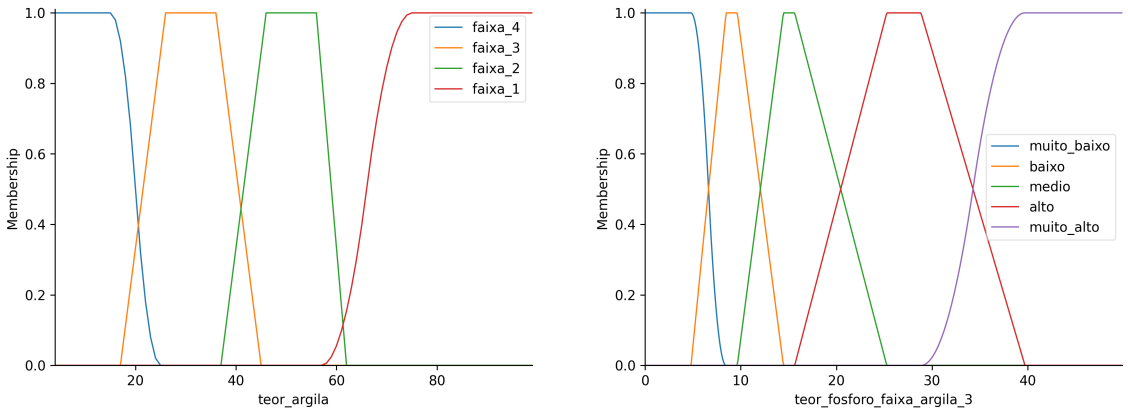
Para obter os *Antecedents* e *Consequents*, foram desenvolvidas duas funções genéricas que constroem os conjuntos *fuzzy* da variável em questão. No Quadro 11 são mostrados trechos das duas funções, onde pode ser observado que um dos parâmetros de entrada é a tabela clássica do MACSCRS que se deseja *fuzzyficar*. Neste exemplo foi utilizada a *tabela_fosforo_grupo_1_classe_3* do Quadro 8. Essa característica demonstra a generalização das funções para todos os cultivos. A chamada para essas funções pode ser observada no Quadro 10, na linha 16.

Quadro 11. Criação do sistema *fuzzy*.

```
    [...]
    def fuzzyfica_antecedent(
        tabela_classica: Tabela,
        universo_fuzzy: NDArray[Union[np.signedinteger, np.floating]],
        nome_antecedent: str,
        list_classificacoes: list[str],
        porcentagem_plato: float = 0.2, ):
        linhas = tabela_classica.linhas
        antecedent = control.Antecedent(universo_fuzzy,
↪ nome_antecedent)
        [...]
    def fuzzyfica_consequent(
        tabela_saida: Tabela,
        universo_fuzzy: NDArray[np.signedinteger],
        nome_saida: str,
        list_classificacoes: list[str],
        suavidade: float = 0.2, ):
        linhas = tabela_saida.linhas
        potassio_por_ha = control.Consequent(universo_fuzzy,
↪ nome_saida)
        [...]
    [...]

```

A lógica que efetivamente constrói os conjuntos foi omitida do Quadro 11 por ser muito extensa, mas o processo se resume a construir, dentro do *Antecedent* ou *Consequent*, um conjunto *fuzzy* para cada faixa da tabela original, como pode ser observado nas Figuras 6(a) (faixa 1 à faixa 4) e 6(b) (*Muito Baixo à Muito Alto*).



(a) Conjuntos *fuzzy* do *Antecedent teor_argila*.

(b) Conjuntos *fuzzy* do *Antecedent teor_potassio_faixa_argila_3*.

Figura 6. Conjuntos *fuzzy* dos *Antecedents*.

A etapa de *defuzzyficação* é a responsável por converter o resultado difuso, obtido após a aplicação das regras *fuzzy*, em um valor numérico único que representa a recomendação final. Na função *calcula_fosforo_fuzzy*, apresentada no Quadro 12, o sistema *fuzzy* é instanciado por meio da função *build_sistema_fuzzy* e, em seguida, recebe como

entrada os valores de argila e fósforo do solo. Esses valores são atribuídos às variáveis de entrada do sistema (*Antecedents*) e o método *compute()* realiza todo o processo de inferência *fuzzy*, considerando as pertinências e regras definidas anteriormente. O resultado numérico é então obtido a partir do *Consequent fosforo_saida*, representando a recomendação de fósforo por hectare, já *defuzzyficada*.

Quadro 12. *Defuzzyficação.*

```
    def calcula_fosforo_fuzzy(
        tabela_saida_classic: Tabela,
        tabelas_entradas_classic: list[Tabela],
        argila_entrada: float,
        fosforo_entrada: float,
    ) -> float:
        sistema_fuzzy = build_sistema_fuzzy(
            tabela_classica_saida=tabela_saida_classic,
            tabela_classica_entrada=tabelas_entradas_classic, )
        fosforo_sim = sistema_fuzzy.sistema
        fosforo_sim.input["teor_argila"] = round(argila_entrada, 2)
        fosforo_sim.input["teor_fosforo_faixa_argila_1"] =
↪ round(fosforo_entrada, 2)
        fosforo_sim.input["teor_fosforo_faixa_argila_2"] =
↪ round(fosforo_entrada, 2)
        fosforo_sim.input["teor_fosforo_faixa_argila_3"] =
↪ round(fosforo_entrada, 2)
        fosforo_sim.input["teor_fosforo_faixa_argila_4"] =
↪ round(fosforo_entrada, 2)
        fosforo_sim.compute()
        resultado = fosforo_sim.output["fosforo_saida"]
    [...]

```

Foi utilizado o método do centro de massa para *defuzzyficar* o sistema, que é o método padrão da biblioteca *sk-fuzzy*. A Figura 7 demonstra visualmente como é feito esse cálculo, para o *Consequent fosforo_saida*.

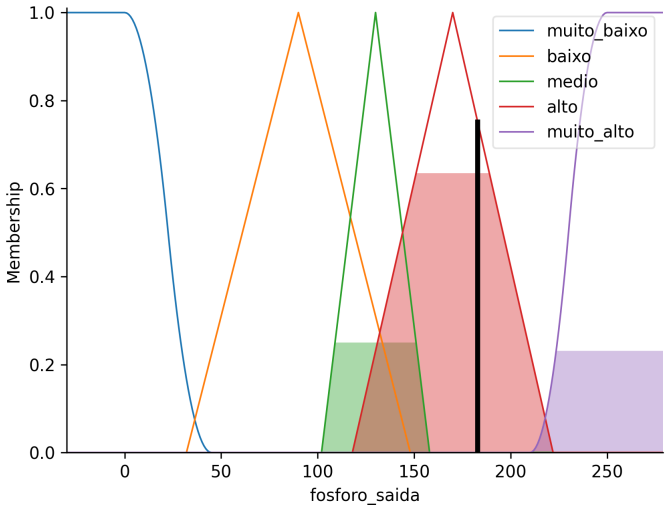


Figura 7. Centro de massa do *Consequent fosforo_saida.*

Todo o procedimento apresentado para o fósforo foi aplicado de forma análoga para o potássio.

3.8. Apresentação do sistema

A Figura 8 apresenta a tela inicial do sistema, que serve como ponto de partida para a navegação entre os módulos. Nela, destaca-se o menu lateral à esquerda, que organiza o acesso às principais funcionalidades da plataforma: *Produtores*, *Propriedades*, *Laudos*, *Recomendações*, *Laboratórios* e *Cultivos*. Essa estrutura garante que o usuário possa acessar rapidamente qualquer módulo desejado, mantendo clareza e consistência na interação com a interface.

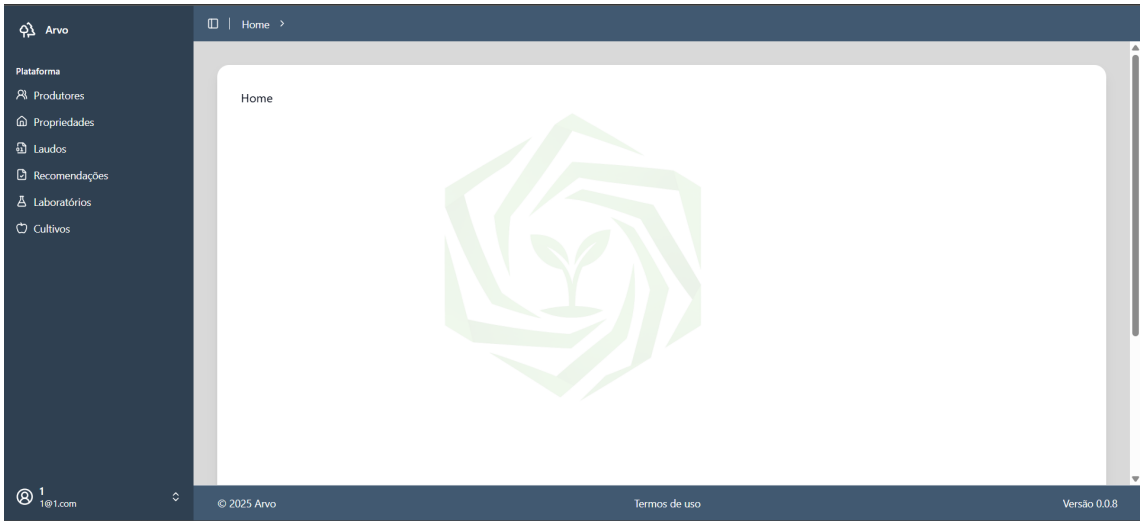


Figura 8. Tela inicial do sistema.

Cada opção do menu leva a uma interface com a lista de entidades correspondentes e alguns de seus atributos. A Figura 9 apresenta um exemplo desta lista, mostrando as propriedades cadastradas, com opções de visualização, edição e exclusão.

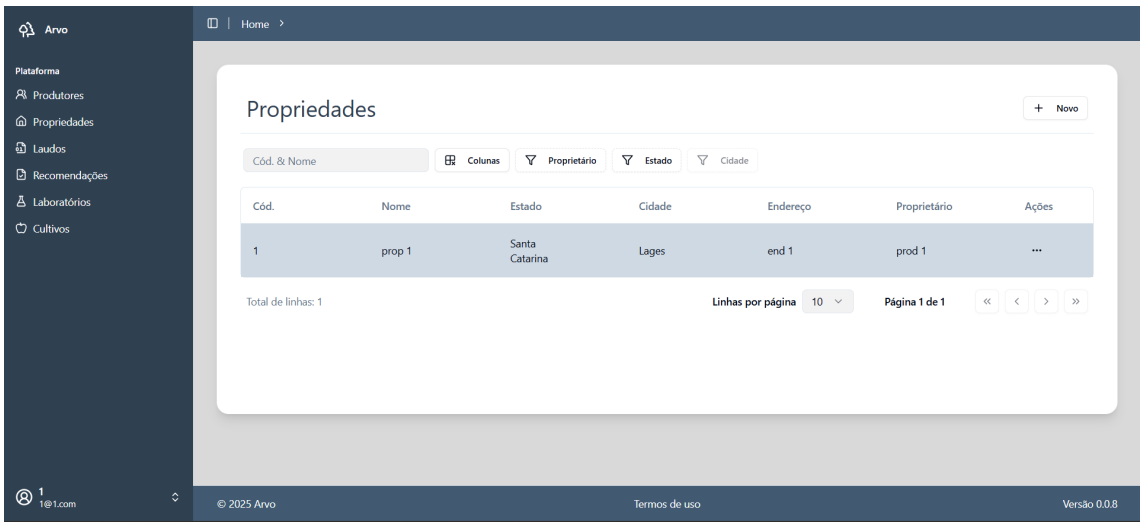


Figura 9. Tela de listagem de propriedades.

Em cada uma dessas telas de listagem, há um botão que leva ao cadastro de uma nova entidade, mostrado na Figura 10, que apresenta a interface de criação de propriedades. Os campos do formulário são reativos e validam os dados antes do envio ao *back-end*. Os demais cadastros simples do sistema, como os de produtores, cidades e estados, seguem a mesma lógica de criação, edição e consulta apresentada no cadastro de propriedades. Por essa razão, eles não são detalhados, evitando redundâncias e mantendo o foco nas funcionalidades mais complexas e relevantes do sistema.

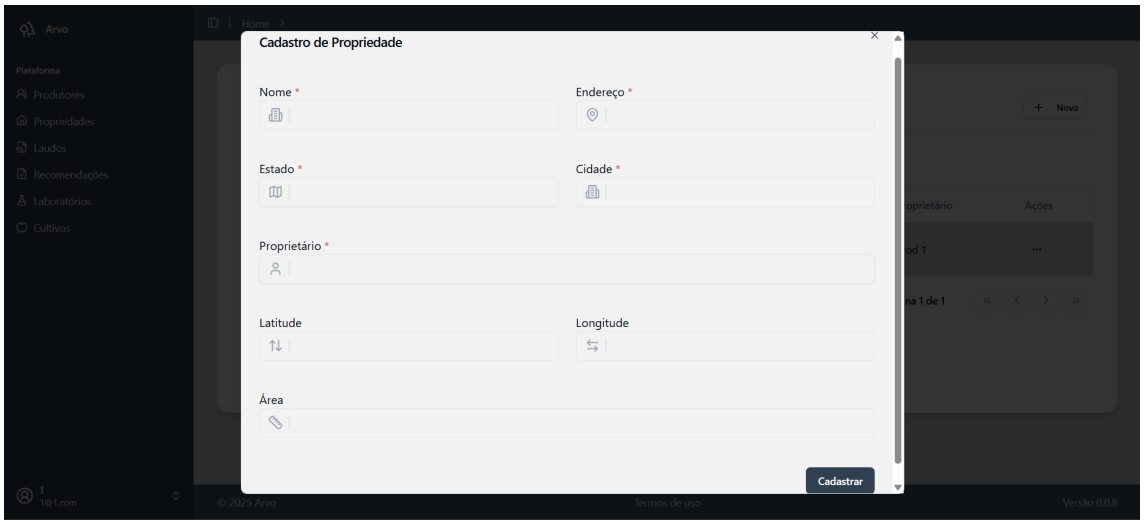


Figura 10. Tela de cadastro de propriedades.

Através do menu *Laudos*, o usuário acessa a interface de listagem de laudos, apresentada na Figura 11. Nela, é possível visualizar os laudos já cadastrados e, ao lado de cada laudo, no botão de *Ações*, editar as informações do laudo e das amostras associadas.

O cadastro de um novo laudo é mostrado na Figura 12. Nessa interface, o usuário registra os resultados de análises de uma propriedade específica. A tela inicial permite preencher dados como identificação da propriedade, data de coleta e laboratório responsável. Ao clicar em *Avançar*, o sistema direciona para o cadastro das amostras correspondentes.

O cadastro de amostras, apresentado na Figura 13, permite registrar informações como a origem da amostra, data de coleta, tipo de cultivo e dados laboratoriais. Esse cadastro é fundamental para a análise do solo e pode ser feito manualmente ou importando diretamente dados dos arquivos PDF com os laudos emitidos por laboratórios cadastrados.

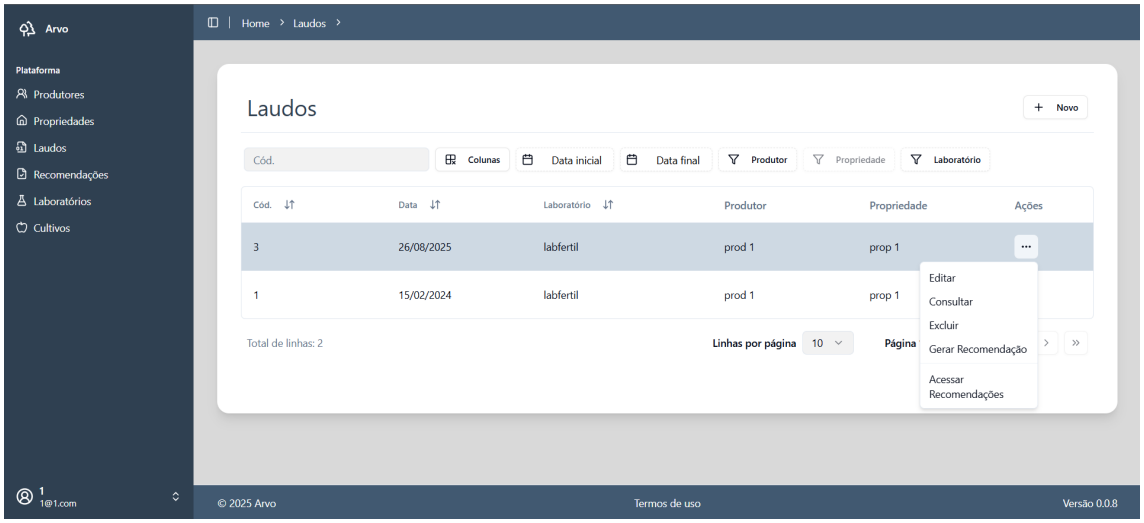


Figura 11. Tela de listagem de laudos.

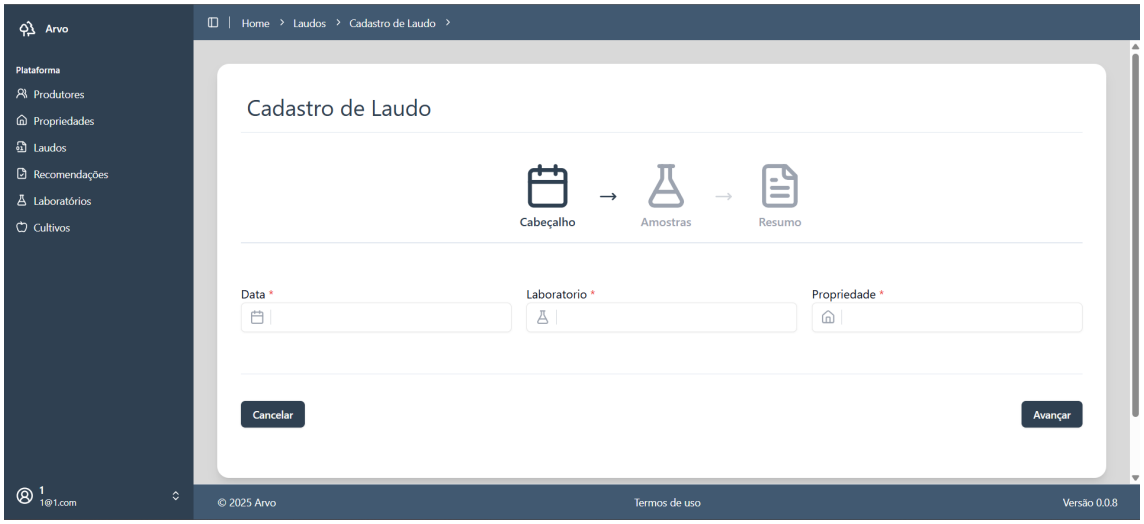


Figura 12. Tela de cadastro de laudos.

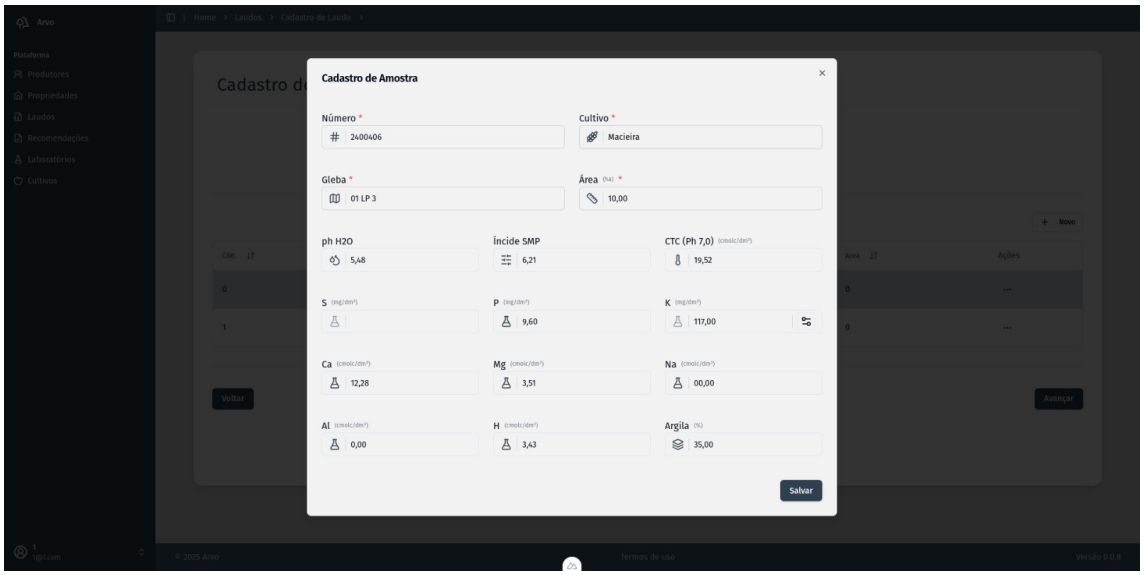


Figura 13. Tela de cadastro de amostras.

Para gerar recomendações, o usuário acessa o menu *Recomendações*. A Figura 14 mostra a lista de recomendações já calculadas. Ao selecionar a opção *Novo*, a interface de emissão de recomendações é exibida (Figura 15), onde o usuário insere informações, como data, profundidade do solo, tipo de adubação e calagem, além de selecionar o laudo e as amostras que serão consideradas na recomendação.

Em seguida, o sistema (Figura 16) apresenta o resultado final das recomendações, permitindo visualizar a recomendação realizada.

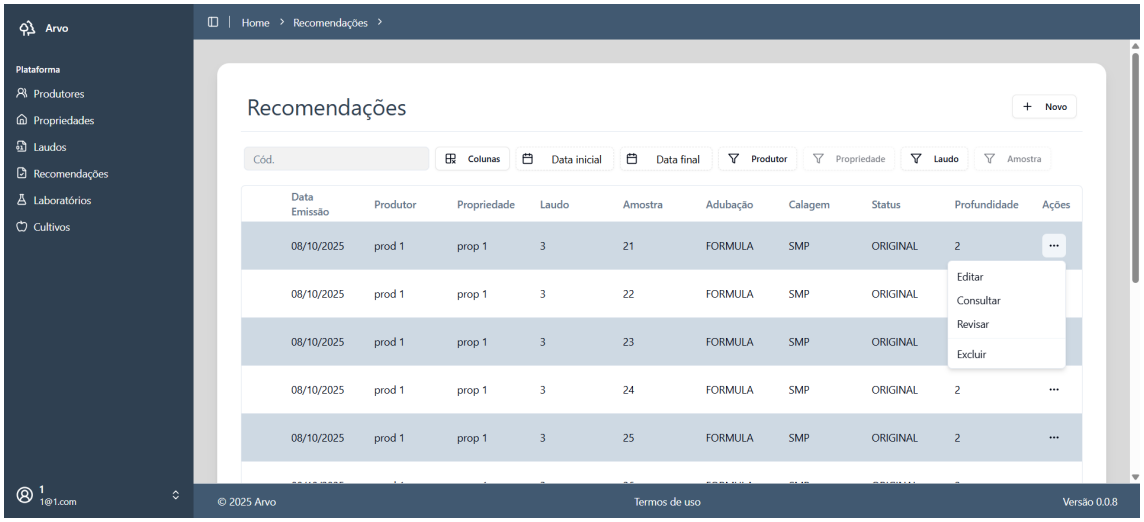


Figura 14. Tela de listagem de recomendações.

Figura 15. Tela com o resultado da recomendação.

Figura 16. Tela com o resultado da recomendação.

4. Testes e resultados

Nesta seção são apresentados os testes realizados com o sistema de recomendação de adubação e calagem desenvolvido, com o objetivo de verificar se os resultados gerados pelo método de tabelas do sistema estão corretos, assim como compará-los com os resultados gerados pelo módulo que usa lógica *fuzzy*.

4.1. Recomendação pelas tabelas do MACSCRS

O primeiro teste realizado visa verificar se os resultados gerados pelo sistema utilizando as tabelas do MACSCRS (Sociedade Brasileira de Ciência do Solo, 2016) são compatíveis com os resultados obtidos por meio de consulta direta ao manual. Para isto, foram analisadas quatro amostras reais: duas de gramíneas de estação fria (IDs 1 e 2) e duas de macieiras (IDs 3 e 4). Os dados de entrada incluem pH, índice SMP, teores de fósforo (P), potássio (K), cálcio (Ca), magnésio (Mg), sódio (Na), Capacidade de Troca Catiônica (CTC) e porcentagem de argila.

O primeiro passo foi comparar as recomendações de calagem segundo o método do Índice SMP. Observa-se que, para todas as amostras, o valor recomendado pelo sistema é exatamente igual ao valor obtido pelo cálculo manual para o respectivo $phsmp$. As amostras de gramíneas (IDs 1 e 2) apresentam doses maiores devido ao pH inicial mais baixo, enquanto as macieiras (IDs 3 e 4) apresentam necessidade menor, o que está coerente com as faixas do manual para esse método (Tabela 8).

Tabela 8. Comparativo de calagem (método Índice SMP)

ID	Dados de Entrada		Recomendação (t/ha)	
	pH H_2O	Índice SMP	Manual	Sistema Arvo
1	4,70	4,71	13,30	13,30
2	5,22	5,25	8,30	8,30
3	5,48	6,21	3,70	3,70
4	5,38	6,16	3,70	3,70

Legenda: IDs 1-2 (Gramíneas); IDs 3-4 (Macieiras).

Em seguida, a Tabela 9 apresenta a calagem pelo método da Saturação por Bases. Novamente, os resultados do sistema coincidem exatamente com os valores manuais para todas as amostras. As gramíneas, com maior acidez relativa, exigem doses mais elevadas, enquanto as macieiras exigem doses muito menores, refletindo diretamente a soma de bases e a CTC de cada uma.

Tabela 9. Comparativo de calagem (método Saturação por Bases)

ID	Dados de Entrada (do Laudo)					Recomendação (t/ha)	
	K (mg/dm³)	Ca (cmolc/dm³)	Mg (cmolc/dm³)	Na (mg/dm³)	CTC (pH 7,0)	Manual	Sistema Arvo
1	97	2,96	1,13	0	23,52	13,30	13,30
2	121	3,58	1,70	0	15,91	6,34	6,34
3	117	12,28	3,51	0	19,52	0,50	0,50
4	151	12,04	5,67	0	21,73	0,37	0,37

Legenda: IDs 1-2 (Gramíneas); IDs 3-4 (Macieiras).

As recomendações de fósforo (P_2O_5) estão apresentadas na Tabela 10. Para as quatro amostras, o sistema reproduziu exatamente os mesmos valores obtidos consultando o manual, sem divergências. Ou seja, a interpretação de fósforo do sistema está totalmente alinhada com o MACSCRS (Sociedade Brasileira de Ciência do Solo, 2016) para os níveis de P e argila observados.

Tabela 10. Comparativo de recomendação de fósforo (P_2O_5)

ID	Dados de Entrada		Recomendação (kg/ha)	
	P (mg/dm³)	Argila (%)	Manual	Sistema Arvo
1	7,1	25	110	110
2	4,2	43	110	110
3	9,6	35	170	170
4	15,6	37	130	130

Legenda: IDs 1-2 (Gramíneas); IDs 3-4 (Macieiras).

Por fim, a Tabela 11 mostra a recomendação de potássio (K_2O). Também neste caso, os valores gerados pelo sistema coincidem integralmente com os valores do manual para todas as amostras, demonstrando correspondência direta entre o cálculo automatizado e o procedimento tradicional.

Tabela 11. Comparativo de recomendação de potássio (K_2O)

	Dados de Entrada		Recomendação (kg/ha)	
ID	**K (mg/dm³)**	**CTC (pH 7,0)**	**Manual**	**Sistema Arvo**
1	97	23,52	90	90
2	121	15,91	60	60
3	117	19,52	60	60
4	151	21,73	30	30

Legenda: IDs 1-2 (Gramíneas); IDs 3-4 (Macieiras).

De forma geral, a comparação evidencia que o sistema está consistente com as tabelas oficiais do MACSCRS (Sociedade Brasileira de Ciência do Solo, 2016), reproduzindo corretamente as recomendações em todos os nutrientes avaliados (calagem por SMP, calagem por saturação, fósforo e potássio). Isso confirma a acurácia das rotinas de cálculo implementadas e valida o comportamento do sistema frente a dados reais de solo.

4.2. Recomendação pelas tabelas *fuzzyficadas*

O objetivo principal desta etapa dos testes foi verificar o funcionamento do sistema de recomendação baseado em lógica *fuzzy* diante de diferentes condições de solo. A intenção não foi apenas avaliar a consistência das recomendações geradas, mas, também, garantir que o modelo respondesse adequadamente às variações dos atributos selecionados, simulando cenários próximos da realidade agrícola.

Para isso, foram definidos intervalos de valores para os principais atributos químicos e físicos do solo, a partir das faixas apresentadas na Seção 2. Os valores foram gerados de forma automática e fictícia, permitindo explorar diferentes combinações possíveis entre os elementos analisados. Dessa forma, buscou-se avaliar como pequenas variações nas condições do solo influenciam a saída do sistema, assegurando que as recomendações permaneçam coerentes.

Os parâmetros escolhidos para a avaliação foram: fósforo (P), potássio (K), argila e Capacidade de Troca Catiônica (CTC). A tabela 12 apresenta o intervalo dos valores testados para cada uma das variáveis, assim como o incremento realizado em cada uma delas.

Tabela 12. Estrutura de valores utilizada nos testes

Elemento	Valor mínimo	Valor máximo	Incremento
Fósforo (P)	1,0	72,0	0,1
Potássio (K)	1,0	330,0	1,0
Argila	1,0	72,0	1,0
CTC	6,0	36,0	0,2

Os testes foram conduzidos por meio de *scripts* desenvolvidos em *Python*, responsáveis por executar iterações sistemáticas sobre os intervalos definidos na Tabela 12. Inicialmente, estabeleceu-se um valor fixo de argila, sobre o qual foram percorridos todos os valores possíveis de fósforo (P). Após completar o intervalo de P, o valor de argila era incrementado conforme o passo definido, repetindo o processo até atingir o limite máximo. Para cada combinação das variáveis de entrada (argila e fósforo), o sistema gerou simultaneamente uma recomendação de necessidade de fósforo com base nas tabelas originais do MACSCRS e outra utilizando as tabelas *fuzzyficadas*. O mesmo procedimento foi adotado para os parâmetros necessários à recomendação de potássio (CTC e K). Todas as recomendações foram registradas em planilhas, associando as variáveis de entrada às respectivas saídas de cada abordagem (convencional e *fuzzyficada*).

A partir das planilhas geradas, foram elaborados gráficos que facilitam a visualização da diferença dos resultados obtidos pelo sistema de recomendação *fuzzy* e aqueles derivados das tabelas dos manuais de adubação convencionais. A Figura 17 demonstra um exemplo de um desses gráficos, nesse caso são apresentados os dados para os diferentes níveis de P com teor de argila fixo em 30%. Nota-se no gráfico que a linha dos resultados *fuzzy* (laranja), conforme aumenta o teor de P, possui uma queda mais suave se comparada com a linha das tabelas convencionais (azul), onde são perceptíveis as mudanças bruscas nas transições das classes de P_2O_5.

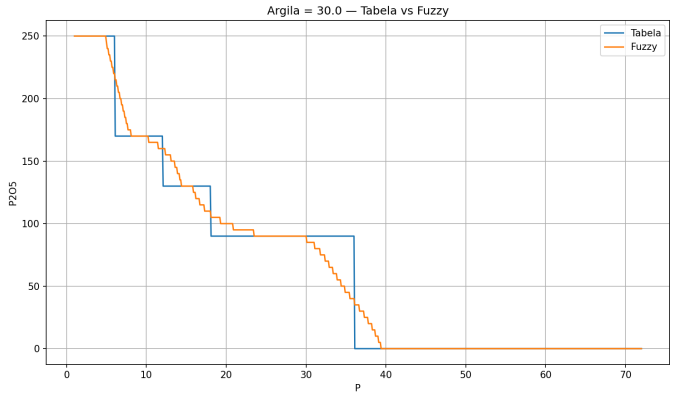


Figura 17. Gráfico comparando recomendações pelas tabelas originais e pelas tabelas *fuzzyficadas*.

Como foram utilizadas duas variáveis de entrada (teor de argila e teor de P) e uma de saída (necessidade de P_2O_5), gráficos em três dimensões foram criados para visuali-

zar todos os dados, que podem ser observados na Figura 18. Analisando estes gráficos, observa-se nitidamente que houve uma suavização nos limites na abordagem *fuzzy*, pois a sua linha se mantém mais contínua, se comparada à linha da tabela original, mesmo assim o *fuzzy* não se afastou da tendência da tabela original, preservando a identidade original das tabelas de início.

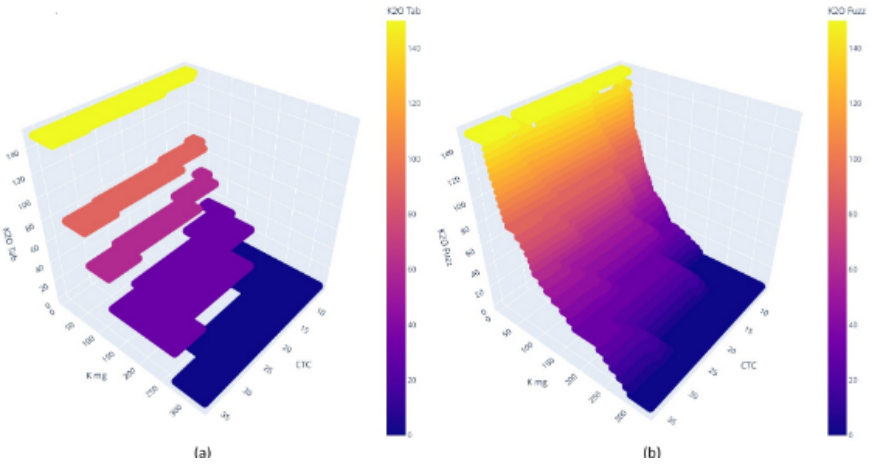


Figura 18. Gráficos 3D comparando recomendações pelas tabelas originais e pelas tabelas *fuzzyficadas*.

Além disso, é importante destacar que a suavização promovida pela abordagem *fuzzy* não ocorre apenas ao longo do eixo correspondente ao fósforo (P) ou ao potássio (K), mas também se estende às variáveis complementares (teor de argila e CTC). Embora essa suavização nos eixos de argila e CTC não seja tão evidente nos gráficos 3D, ela está presente de forma consistente, indicando que a transição entre as classes dessas variáveis também se torna mais contínua e coerente no modelo *fuzzyficado*. Essa característica reforça a capacidade do sistema de representar de maneira mais gradual as relações entre os parâmetros do solo e as recomendações de adubação.

5. Conclusão

O desenvolvimento deste trabalho permitiu demonstrar a viabilidade de automatizar recomendações de adubação e calagem a partir das diretrizes do MACSCRS. Os resultados evidenciaram que o sistema é capaz de reproduzir, de maneira consistente, as recomendações obtidas por consulta direta ao manual, evitando erros decorrentes da interpretação humana. Assim, a aplicação computacional proposta consolida uma forma padronizada, reproduzível e rastreável de gerar recomendações, contribuindo para a redução de inconsistências operacionais no processo de tomada de decisão em fertilidade de solos.

Além disso, com base nos testes da seção 4.2, percebe-se que a incorporação da lógica *fuzzy* demonstrou eficiência em suavizar as transições abruptas presentes nas recomendações do método tradicional. Enquanto as tabelas do manual oficial apresentam variações discretas e descontinuidades perceptíveis entre faixas de atributos do solo, o modelo *fuzzy* proporcionou uma resposta contínua e mais coerente com a variação gradual das condições reais. Essa característica permitiu um resultado com curvas mais estáveis e uniformes, conforme evidenciado nos gráficos comparativos. Dessa forma, a lógica

fuzzy demonstrou ser uma técnica promissora para buscar um aumento na eficiência das recomendações de adubação.

Além dos resultados técnicos alcançados, destaca-se o desenvolvimento e a disponibilização do sistema web, projetado para ser acessível a partir de qualquer dispositivo com conexão à Internet, permitindo que agrônomos e demais profissionais utilizem de forma prática as funcionalidades implementadas. O sistema possibilita gerar recomendações tanto pelo método tradicional do manual oficial quanto pela abordagem *fuzzy*, oferecendo uma ferramenta comparativa que amplia a compreensão sobre o comportamento de cada modelo.

Como trabalhos futuros, propõe-se a realização de novos testes voltados à calibração dos parâmetros dos conjuntos *fuzzy*, buscando otimizar ainda mais a precisão das inferências e a adequação dos limites de pertinência às condições reais de campo. Além disso, recomenda-se a condução de experimentos práticos em diferentes culturas e tipos de solo, a fim de avaliar o impacto financeiro e agronômico das recomendações geradas pelo sistema no módulo *fuzzy*, verificando sua influência direta sobre o crescimento das plantas e o retorno econômico proporcionado pelas distintas doses de adubação sugeridas.

Referências

Bobadilla, J., Ortega, F., Hernando, A., e Gutiérrez, A. (2013). Recommender systems survey. *Knowledge-Based Systems*, 46:109–132.

Bönisch, S., Lopes-Assad, M. L., Monteiro, A. M. V., e Câmara, G. (2004). Representação e propagação de incertezas em dados de solo: II – atributos numéricos. *Miscellaneous*, 28(1):33–47.

Carvalho, F. S., Almeida, R. T., e Gomes, L. A. (2019). Resposta do milho a doses de nitrogênio em cobertura: comparação entre ureia e sulfato de amônio. *Pesquisa Agropecuária Brasileira*, 54(8):e01234.

Choueri, M. (2024). *Modelos fuzzy da cultura do milho em resposta às diferentes doses e fontes de nitrogênio em cobertura*. Tese de doutorado, Faculdade de Ciências e Engenharia, Universidade Estadual Paulista "Júlio de Mesquita Filho", Tupã, SP, Brasil.

Corrêa, J. C., Rebellatto, A., Grohskopf, M. A., Cassol, P. C., Hentz, P., e Rigo, A. Z. (2015). Fertilidade do solo e produtividade agrícola com aplicação de fertilizantes organominerais ou minerais nas formas sólidas e fluidas. *Pesquisa Agropecuária Brasileira*, 50(9):769–778.

Godinho, E., Marçal, R., Oliveira, A., e Galvão, V. (2022). Lógica fuzzy aplicada na produtividade da cenoura. *ResearchGate*.

Klir, G. J. e Yuan, B. (1995). *Fuzzy Sets and Fuzzy Logic: Theory and Applications*. Prentice Hall.

Melo, G. W. (2003). Sistema de produção de pêssego de mesa na região da serra gaúcha. *Sistema de Produção*. Versão eletrônica. Disponível em: https://www.infoteca.cnptia.embrapa.br/infoteca/handle/doc/542650.

Melo, G. W. B. d. e Brunetto, G. (2016). Adubação e manejo do solo. In Hoffmann, A., Silveira, S. V. d., e Garrido, L. d. R., editors, *Produção integrada de uva para processamento: fertilidade e manejo do solo e da água*, volume 2, chapter 1, pages 9–16. Embrapa, Brasília, DF. Acesso aberto. Disponível em: https://www.alice.cnptia.embrapa.br/handle/doc/1046462.

Mendel, J. M. (1995). Fuzzy logic systems for engineering: a tutorial. *Proceedings of the IEEE*, 83(3):345–377.

Novais, R. F., Alvarez, V. H., Barros, N. F., Fontes, R. L. F., Cantarutti, R. B., e Neves, J. C. L. (2007). *Fertilidade do solo*. Sociedade Brasileira de Ciência do Solo, Viçosa, MG.

Papadopoulos, K., Kalivas, D., e Hatzichristos, T. (2011). Decision support system for nitrogen fertilization using fuzzy theory. *Computers and Electronics in Agriculture*, 79(2):178–188.

Pereira, L. A. C. e Cardoso, V. M. (2025). How dependent is brazilian agriculture on fertilizer imports? *Agro in Data — Insper*. Published 07/08/2025.

Ross, T. J. (2010). *Fuzzy logic with engineering applications*. Wiley, Chichester, UK, 3rd edition.

Santos, A. S. e Almeida, L. L. (2017). Desafios e inovações na gestão sustentável da fertilização e correção do solo no Brasil. *Revista Brasileira de Ciência do Solo*, 41:1–16.

Santos, K. E. L., Bernardi, A. C. d. C., Bettiol, G. M., e Crestana, S. (2017). Geostatística e geoprocessamento na tomada de decisão do uso de insumos em uma pastagem. *Miscellaneous*, 11(3):294–307.

Shanthi, R. e Gomathi, P. (2015). A study of fuzzy logic applications in precision agriculture. *International Journal of Advanced Research in Computer Science and Software Engineering*, 5(1):647–650.

Shoemaker, H. E., McLean, E. O., e Pratt, P. F. (1961). Buffer methods for determining lime requirement of soils with appreciable amounts of extractable aluminum. *Soil Science Society of America Proceedings*, 25(4):274–277.

Silva, E. F. d. e Lima, J. R. F. (2009). Lógica fuzzy no mapeamento de variáveis indicadoras de fertilidade do solo. *Chilean Journal of Agricultural Research*, 69(3):418–426.

Sociedade Brasileira de Ciência do Solo (2016). *Manual de calagem e adubação para os Estados do Rio Grande do Sul e de Santa Catarina*. Comissão de Química e Fertilidade do Solo – RS/SC, Núcleo Regional Sul.

Torres, A., Nieto, J., e Aguilar, R. (2006). Medical diagnosis using a fuzzy logic approach. *International Journal of Approximate Reasoning*, 41(2):163–184.

Zadeh, L. A. (1965). Fuzzy sets. *Information and Control*, 8(3):338–353.