UFRJ

Instituto de Matemática

UMA TECNOLOGIA PARA REDAÇÃO MATEMÁTICA E SEU USO NA ELABORAÇÃO DE UM CURSO DE ÁLGEBRA LINEAR

Dissertação de Mestrado

Rodrigo Gomes Devolder

Orientador: Luiz Carlos Guimarães

Rio de Janeiro
Agosto de 2012

UMA TECNOLOGIA PARA REDAÇÃO MATEMÁTICA E SEU USO NA ELABORAÇÃO DE UM CURSO DE ÁLGEBRA LINEAR

Rodrigo Gomes Devolder

DISSERTAÇÃO SUBMETIDA AO CORPO DOCENTE DO INSTITUTO DE MATEMÁTICA DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS À OBTENÇÃO DO GRAU DE MESTRE EM CIÊNCIAS EM ENSINO DA MATEMÁTICA.

Aprovada por:

Prof. Luiz Carlos Guimarães, Ph.D., IM/UFRJ (Orientador)

Prof. Yuriko Yamamoto Baldin, D.Sc., DM/UFSCar

Prof. Marta Feijó Barroso, D.Sc., IF/UFRJ

RIO DE JANEIRO, RJ – BRASIL
AGOSTO DE 2012

DEVOLDER, RODRIGO GOMES

Uma Tecnologia para Redação Matemática e seu Uso na Elaboração de um Curso de Álgebra Linear/Rodrigo Gomes Devolder. – Rio de Janeiro: UFRJ/IM, 2012.

VII, 46 p., 29,7cm.

Orientador: Luiz Carlos Guimarães

Dissertação (mestrado) – UFRJ/IM/PEMAT, 2012.

Referências Bibliográficas: p. 42–43.

1. Sistema de Gestão da Aprendizagem. 2. Sistema de Computação Algébrica. 3. Álgebra Linear.

I. Guimarães, Luiz Carlos. II. Universidade Federal do Rio de Janeiro, Instituto de Matemática, Programa de Ensino da Matemática. III. Título.

Resumo da Dissertação apresentada ao IM/UFRJ como parte dos requisitos necessários à obtenção do grau de Mestre em Ciências (M.Sc.)

UMA TECNOLOGIA PARA REDAÇÃO MATEMÁTICA E SEU USO NA ELABORAÇÃO DE UM CURSO DE ÁLGEBRA LINEAR

Rodrigo Gomes Devolder

Agosto/2012

Orientador: Luiz Carlos Guimarães

Programa: Ensino da Matemática

Apresentamos neste trabalho uma proposta de tecnologia para redação de textos matemáticos desenvolvida no LIMC/UFRJ e uma análise de viabilidade de uso desta tecnologia em cursos regulares de matemática. Esta tecnologia integra duas plataformas computacionais: um Sistema de Gestão da Aprendizagem (LMS) e um Sistema de Computação Algébrica (CAS). A integração, através de uma sintaxe de redação, permite introduzir expressões matemáticas a serem processadas e exibidas sempre que o documento é trazido à visualização. A reunião de textos que utilizam esta tecnologia formam um acervo acessível aos participantes inscritos em um dado curso no LMS, proporcionando ao professor um poderoso complemento didático e aos estudantes um espaço amigável para discutir conceitos e abordar conteúdos próprios à matemática. Para verificação e análise da viabilidade do uso da tecnologia desenvolvida e de sua implementação, acompanhamos a sua aplicação e suas contribuições em um curso regular de Álgebra Linear II.

Abstract of Dissertation presented to IM/UFRJ as a partial fulfillment of the requirements for the degree of Master of Science (M.Sc.)

A TECHNOLOGY FOR MATH WRITING AND ITS USAGE IN A LINEAR ALGEBRA COURSE

Rodrigo Gomes Devolder

August/2012

Advisor: Luiz Carlos Guimarães

Department: Mathematics Teaching

Sumário

Lista de Figuras

Capítulo 1

Introdução

1.1 A Proposta da Dissertação

Esta dissertação de mestrado propõe uma tecnologia para redação de textos matemáticos e apresenta uma descrição funcional de sua implementação através da integração de duas plataformas computacionais: um Sistema de Gestão da Aprendizagem (LMS) e um Sistema de Computação Algébrica (CAS). Também analisamos a viabilidade de uso da tecnologia proposta em cursos regulares de matemática, tanto na sala de aula quanto em interações extraclasse, proporcionando ao professor um poderoso complemento didático e aos estudantes um espaço amigável para discutir conceitos e abordar conteúdos próprios à matemática.

Para a implementação desta tecnologia, foi concebida uma sintaxe de redação que permite, em um documento de texto, introduzir expressões matemáticas a serem processadas e exibidas na tela do computador sempre que o documento é trazido à visualização. Esta sintaxe prevê a inclusão de comandos a serem processados pelo CAS, permitindo ao usuário explorar toda a potencialidade oferecida por um software deste porte, ao mesmo tempo em que usufrui das conhecidas facilidades disponíveis em uma plataforma LMS.

Utilizamos a plataforma Moodle (DOUGIAMAS, 2003, [3]; MOORE, 2003, [11]) por ser hoje o LMS de grande uso, em conjunto com o Maxima (SOUZA *et al*, 2004, [13]), um CAS bastante difundido na comunidade acadêmica. É importante assinalar que as escolhas do Moodle e do Maxima não foram arbitrárias: estes programas são largamente difundidos na comunidade acadêmica, são relativamente fáceis de instalar, alterar e manter, além de possuírem código aberto. Ressaltamos, entretanto, que a tecnologia que estamos propondo poderia ser implementada através da integração de outros programas de naturezas similares ao Moodle e ao Maxima.

A interação com o CAS possibilita a inserção de conteúdo matemático dinâmico em um documento de texto, da seguinte forma: expressões matemáticas envolvidas entre cifrões no texto sendo editado (em uma sintaxe análoga à do processador de textos LaTeX) são analisadas, com o objetivo de identificar trechos que exijam processamento matemático. Estas porções são destacadas do texto e enviadas ao CAS, que realiza os cálculos necessários e emite as respostas requeridas. Na exibição, as respostas do CAS são inseridas nos locais apropriados do texto, onde originalmente havia as expressões.

Os documentos de texto contendo expressões matemáticas, incluindo aí o que denominamos *scripts*, formam um acervo didático que fica acessível aos participantes inscritos em um dado curso no LMS. Um *script*, conforme definiremos no Capítulo 2, nada mais é do que um documento que contém grupos de comandos a serem processados pelo CAS, em forma algorítmica, e cujos resultados serão visualizados no momento da exibição final do documento na tela. O usuário pode alterar dados e o próprio algoritmo de um *script* quantas vezes julgar necessário e reprocessar o documento *online*. Esta capacidade de reprocessar o documento a qualquer tempo mostrou-se um poderoso recurso didático, tanto para uso em sala de aula quanto para uso extraclasse por parte dos alunos, que podem, a partir de um script disponibilizado pelo professor, produzir alterações e, com isto, explorar variações de cenários distintos sobre um mesmo tópico da disciplina.

Uma das vantagens principais desta ferramenta que propomos é a singularidade desta integração entre um LMS e um CAS: por um lado, agregamos ao CAS uma interface amigável de uso e, por outro, acrescentamos ao LMS a capacidade de processar e exibir, com qualidade, conteúdo matemático.

Este ganho é potencializado na medida em que o uso remoto via internet e as possibilidades de replicar este modelo em múltiplos cursos permitem o acesso em larga escala por parte da comunidade a uma ferramenta para comunicação efetiva e processamento de conteúdos matemáticos.

A tecnologia e sua implementação, validadas neste trabalho, constituem mais um elemento que faz parte de um conjunto de produtos desenvolvidos no Laboratório de Pesquisa e Desenvolvimento em Ensino de Matemática e Ciências – LIMC/UFRJ [1] em uma perspectiva que visa a ampliar o acesso ao ensino de matemática ao mesmo tempo que permite variações didáticas com uso de tecnologia em sala de aula, bem como funcionalidades para sistemas com atuação remota visando a viabilizar o Ensino de Matemática a Distância. Neste contexto, os recursos tecnológicos devem ser projetados para que atuem como mediadores do processo de ensino e aprendizagem criando novas possibilidades a serem exploradas por professores na elaboração de suas aulas, bem como disponibilizando ao aluno ferramentas didáticos que contri-

[1] `www.limc.ufrj.br`

buam efetivamente ao bom entendimento de conceitos fundamentais.

1.2 Justificativa

O desenvolvimento da tecnologia e sua implementação por meio da integração dos ambientes computacionais já citados – o Moodle e o Maxima – originaram-se da observação de uso em sala de aula da plataforma MathMoodle, previamente desenvolvida por nós e descrita por (DEVOLDER, 2008, [2]). O objetivo era integrar um sistema de computação algébrica (CAS) a um sistema de gerenciamento de conteúdos voltados para o ensino e aprendizagem (LMS). Desde a sua versão preliminar, o MathMoodle vem sendo gradualmente introduzido em disciplinas de cunho matemático na Universidade Federal do Rio de Janeiro, tanto como suporte a aulas presenciais quanto para atividades realizadas fora da sala de aula por alunos e monitores.

A primeira disciplina que utilizou o MathMoodle, ainda em sua versão preliminar, foi objeto de pesquisa na dissertação de mestrado de (SILVA, 2009, [12]), que acompanhou durante um semestre um curso de Cálculo II em que o professor incentivou o uso da plataforma propondo atividades complementares. Os alunos e o professor usaram a plataforma em atividades que exploravam meios síncronos e assíncronos do LMS para comunicar conteúdos matemáticos, especialmente por fóruns e chats.

Da análise e observações dos resultados obtidos no trabalho desenvolvido por (SILVA, 2009, [12]) surgiram as primeiras ideias que motivaram o desenvolvimento deste nosso trabalho. Recolhemos as impressões de alunos, do professor da disciplina e do próprio pesquisador, identificando algumas demandas que ocasionaram a reformulação do MathMoodle.

Dois pontos relevantes, observados durante este primeiro experimento, motivaram-nos a redesenhar completamente a plataforma:

- Problemas relacionados a erros ou instabilidades, próprios a sistemas em teste, que acabaram por desviar o foco do conteúdo ensinado;
- A heterogeneidade da sintaxe que havíamos proposto na primeira versão do MathMoodle: os comandos matemáticos e de exibição eram introduzidos no texto sem qualquer delimitação léxica e a distinção entre eles não era clara para o usuário. As dificuldades com a sintaxe acabavam por desviar a atenção dos alunos do conteúdo matemático sendo ensinado, desmotivando-os ao uso da plataforma.

A busca pela solução destes probelmas levou ao aprimoramento da ferramenta que possuíamos, de modo que ela pudesse contribuir efetivamente para o enriquecimento didático das aulas e levasse os alunos a um melhor entendimento de conceitos fundamentais às disciplinas de matemática. Os detalhes acerca dos erros detectados e das correções ao projeto serão brevemente descritos no Capítulo 2.

A implementação atual, fruto de uma total revisão, descrita no Capítulo 2, baseia-se no fato de a edição de textos no Moodle estar concentrada em um único módulo: o editor TinyMCE[2]. Acrescentando funcionalidades a este módulo, tornamo-lo capaz de interagir com o Maxima e processar de forma *online* cálculos matemáticos. Este novo módulo aumentado, que denominamos *MathWriting*, é o cerne da tecnologia que descreveremos neste trabalho. Ele permite, através de uma sintaxe semelhante à do LaTeX, inserir expressões matemáticas em documentos processados pelos demais módulos do Moodle: Chats, Fóruns, Glossários, dentre outros.

Um olhar mais analítico após o processo de redesenho do MathMoodle revelou que tínhamos em mãos não apenas uma integração entre programas existentes, mas sim uma tecnologia de redação matemática, que pode concretizar-se computacionalmente de diferentes maneiras: A integração Moodle-Maxima-Navegador Web, objeto desta dissertação, é apenas uma das possibilidades, mas outras implementações são possíveis. É esta tecnologia e seus componentes que detalharemos no Capítulo 2.

Para verificação e análise da viabilidade do uso da tecnologia desenvolvida e de sua implementação, acompanhamos a sua aplicação em uma disciplina regular de Álgebra Linear II ministrado no Instituto de Matemática da Universidade Federal do Rio de Janeiro. Inicialmente foi feito um estudo das necessidades e demandas do professor da disciplina e uma previsão de aspectos desejáveis que pudessem ser oferecidos pela ferramenta a ser desenvolvida, tanto em sala de aula quanto em atividades extraclasse desenvolvidos no LMS e ainda as ações que deveriam orientar as aulas de monitoria (GUIMARÃES *et al*, 2012, [5]).

Durante o acompanhamento das aulas de Álgebra Linear II, notamos que a disponibilização de scripts em um curso mostrou-se um recurso didático valioso: tanto o professor os utilizava em sala de aula, produzindo uma variada gama de exemplos a fim de ilustrar amplamente um determinado tópico, quanto os alunos eram incentivados, a partir dos scripts oferecidos, a alterá-los e a produzir, a partir deles, os seus próprios scripts.

Esta demanda, inusitada até então, motivou-nos o acréscimo de um novo módulo no Moodle, que pode ser incorporado a qualquer curso: o *Módulo Scripts*. Nele, o professor disponibiliza seus scripts, aos quais todos os participantes do curso

[2] `www.tinymce.com`

têm acesso. Este recurso permite a alunos e monitores paulatinamente dominarem a sintaxe matemática proposta, usufruindo das potencialidades do CAS, sem, no entanto, se verem obrigados a aprender uma quantidade grande de comandos específicos. Num processo de colagem e adaptação, o estudante vai construindo seus próprios scripts a partir de modelos fornecidos pelo professor.

1.3 O Relato do Trabalho

As ações desenvolvidas para a realização deste trabalho iniciaram com a análise dos resultados apresentados na dissertação de mestrado de (SILVA, 2009, [12]). Inicialmente, como desenvolvedores do MathMoodle e MathWriting, acompanhamos como observadores independentes o desenvolvimento do curso de Cálculo II, com o objetivo de coletar informações sobre o comportamento da tecnologia aplicada no caso, anotando os problemas, dificuldades e os aspectos favoráveis ao uso da referida plataforma.

Após a análise destes dados e o redesenho da plataforma, elegemos o curso de Álgebra Linear II, ministrado a turmas de Engenharia, para realizar o experimento de validação. Examinamos, juntamente com o professor, a bibliografia do curso, a fim de identificar pontos chaves do conteúdo programático da disciplina, observando as diversas abordagens da literatura para os conceitos a serem apresentados, de modo a destacar aqueles que receberiam um tratamento especial por meio do desenvolvimento de scripts.

Após a escolha dos conteúdos específicos a serem objeto de análise e das aulas em que utilizaríamos a ferramenta, foram feitos estudos para encontrar a melhor forma de desenvolvê-las. Com a ajuda do professor, esboçamos scripts iniciais que, ao longo das aulas e do uso, foram sendo modificados e ajustados à necessidade do professor e da turma.

O acompanhamento da preparação das aulas pelo professor e a observação destas aulas ministradas durante dois períodos letivos sucessivos contribuíram para a constatação da viabilidade de nossa proposta no planejamento e execução do curso. Foi ainda observado o uso por parte dos alunos, desenvolvendo autonomia para alterar e adequar os referidos scripts em estudos realizados na plataforma, bem como em aulas de monitoria. Participamos ativamente de todo o processo: a preparação dos scripts para as aulas, o acompanhamento das mesmas a implementação de eventuais ajustes e correções necessários ao funcionamento eficiente tanto dos scripts quanto da plataforma computacional.

Durante as aulas foi possível observar e interpretar a relevância que os alunos atribuíram ao uso da ferramenta em aula, que a tornava bem mais dinâmica e atraente devido à riqueza de ilustrações que era possível fornecer sobre cada tópico escolhido. Neste caso buscamos entender aspectos do aprendizado dos alunos, verificando o grau de interesse durante as aulas e a participação ativa em questões conceituais, sendo este enfoque baseado mais numa percepção do pesquisador observador, permitindo uma análise qualitativa do processo desenvolvido em sala de aula e posteriormente nos fóruns do MathMoodle.

A fase final da pesquisa foi de recolhimento de partes do curso que permitiram constatar a viabilidade de nossa proposta. Estas partes fundamentam a afirmação sobre a viabilidade do uso deste tipo de ferramenta neste curso de Álgebra Linear, abrindo possibilidades para futuros desenvolvimentos em outras disciplinas e segmentos do ensino de matemática como, por exemplo, o Ensino Básico.

1.4 Estrutura da Dissertação

Esta dissertação está organizada em quatro capítulos, incluindo esta introdução.

No Capítulo 2, descrevemos os pontos que nos levaram a reformular a ferramenta que possuíamos, bem como a visão analítica sobre a nova realidade que nos permite apresentar as ideias aqui propostas como uma tecnologia de redação matemática, em vez de uma simples integração de ambientes computacionais pré-existentes.

O Capítulo 3 dedica-se à descrição do experimento de validação, no qual a ferramenta foi utilizada em um curso regular de Álgebra Linear II da UFRJ.

O Capítulo 4 conclui o texto, resumindo algumas das principais observações que realizamos ao longo do processo de implementação e validação, além de apontar possibilidades futuras para o aprimoramento da ferramenta e para seu uso em outros contextos, inclusive não universitários.

,

Capítulo 2

A Evolução do MathMoodle

Neste capítulo descreveremos, em um nível apenas funcional, a implementação realizada para a integração Moodle-Maxima que propomos: o MathMoodle. Está, portanto, fora desta proposta de trabalho a abordagem dos aspectos computacionais envolvidos.

Iniciamos com uma breve exposição de algumas das deficiências apresentadas pela versão anterior do MathMoodle, que foram constatadas ao longo de sua utilização na disciplina de Cálculo II, conforme descrito em (SILVA, 2009, [12]). Mostramos como estas deficiências foram sanadas, com a introdução de uma sintaxe mais robusta e coerente, que eliminou as ambiguidades que a versão anterior apresentava.

A criação de dois novos modos de interação com o Maxima – o modo silencioso e o modo literal – fez surgir um modelo (ou, como se diz em inglês, um *template*) para estruturação de documentos com conteúdo matemático. Aos documentos que seguem este modelo denominamos *scripts*. Seu uso foi tão frequente e o modelo tão obviamente vantajoso que tornou-se quase um paradigma de redação e acabou por motivar o acréscimo de um novo módulo específico, destinado a disponibilizar scripts no Moodle.

Por fim, num exercício de abstração, observamos a implementação realizada de um ponto de vista puramente estrutural, identificando e nomeando seus componentes, definindo suas responsabilidades e explicitando sua inter-relação. A estrutura modular que se evidenciou neste processo revelou alguma abrangência, na medida em que é passível de consolidar-se em diferentes cenários computacionais. Um destes cenários é exatamente a integração Moodle-Maxima que descrevemos e validamos; mas outras possibilidades serão ilustradas na última seção deste capítulo.

A esta estrutura modular demos o nome de uma tecnologia para redação matemática, que dá título a esta dissertação.

2.1 A Experiência Anterior com o MathMoodle

O MathMoodle (DEVOLDER, 2008, [2]) foi inicialmente pensado como um sistema computacional que integrava cooperativamente dois ambientes: o LMS Moodle e o CAS Maxima. A ideia central, na época em que foi concebido (2008-2009), era simplesmente acrescentar a alguns módulos do Moodle a capacidade de comunicar conteúdo matemático de forma inteligível, agregando também a possibilidade de avaliar expressões dinamicamente, com o auxílio do CAS. Assim, no âmbito de um curso do Moodle, o participante inscrito poderia, em um fórum de discussões ou num chat, redigir mensagens que contivessem expressões matemáticas, que seriam percebidas pelos demais participantes (no caso de um fórum) ou por seus interlocutores (no caso de um chat) de maneira formatada e legível, como se espera normalmente de textos matemáticos produzidos com qualidade.

Para exemplificar brevemente o uso e a sintaxe, o usuário poderia, em uma mensagem de chat do Moodle, digitar o seguinte texto:

```
A raiz quadrada de x:=9 é !sqrt(x).
```

e o resultado percebido pelo interlocutor seria:

```
A raiz quadrada de x:=9 é 3.
```

O plug-in que desenvolvemos para realizar o processamento dos textos, identificando, extraindo, enviando os trechos matemáticos para processamento no CAS e coletando as respostas, chamou-se MathWriting e era acoplado ao editor de textos HTMLArea, utilizado por quase todos os módulos do Moodle.

O processamento realizado sobre o texto envolvia uma análise sintática rudimentar das frases, em busca de símbolos que denotassem conteúdo matemático. Estes símbolos recaíam em três categorias: operadores aritméticos, funções e primitivas de exibição. Atribuições de valores a variáveis (como no exemplo, `x:=9`) e chamadas de funções (no exemplo, `!sqrt`) eram convertidas para a sintaxe do Maxima e enviadas a ele, a fim de serem processadas; os resultados retornados eram inseridos no lugar das expressões originais.

Ao contrário das funções, que envolviam cálculos matemáticos a serem efetuados, as primitivas de exibição não se traduziam em invocações ao Maxima; destinavam-se tão somente a orientar a "renderização" feita pelo navegador para a produção do documento final na tela.

Apesar de confortável e intuitiva para o redator do texto, a simplicidade desta sintaxe apresentava dois fortes inconvenientes.

Em primeiro lugar, não era muito claro para o usuário quais identificadores referiam-se a funções que correspondiam a comandos do Maxima e quais referiam-se apenas a primitivas de exibição: o emprego dos caracteres `\` e `!`, usados para diferenciar as funções das primitivas de exibição, era sempre motivo de confusão e dúvida.

Um outro ponto mais grave dizia respeito à falta de uma delimitação nítida das expressões que livrasse o reconhecimento sintático de ambiguidades, já que a busca de conteúdo matemático nas frases orientava-se pelos símbolos e pelos nomes de funções e primitivas. Desta forma, a frase

`Hoje é dia 20/8/2012`

seria processada como

$$\texttt{Hoje é dia } \tfrac{20}{8}\texttt{/2012}$$

pois a data seria percebida pela análise sintática como uma expressão envolvendo frações. Era considerada um erro de sintaxe a utilização de duas barras sucessivas, sem a separação por um caractere espaço, como no caso da data. Para exibir corretamente, na antiga sintaxe, era necessário o uso do caractere `'` antes, neste formato:

`Hoje é dia '20/8/2012`

O apóstrofo indicava ao analisador sintático que este trecho não deveria ser processado como fração.

Estes problemas e ambiguidades dificultavam o aprendizado por parte do usuário e os erros deste tipo eram muito frequentes. Como consequência, muitas vezes o foco no conteúdo matemático em questão acabava desviado por conta de erros na sintaxe. A reflexão acerca de como sanar estas deficiências sintáticas e de como estender ao maior número possível de módulos do Moodle a capacidade de armazenamento e processamento de conteúdo matemático, levou-nos a uma reestruturação do MathMoodle, tanto do ponto de vista da sintaxe adotada na escrita dos textos quanto da implementação do MathWriting. A revisão sintática e as novas funcionalidades acrescidas serão descritas na próxima seção.

2.2 O Redesenho e as Novas Funcionalidades

As denominações *MathMoodle* para a ferramenta e *MathWriting* para o plug-in do editor de textos foram mantidas na nova versão, evitando criar novas designações.

A principal mudança requerida, conforme já mencionamos, residia na sintaxe, que, apesar de confortável e simples, apresentava ambiguidades incontornáveis do ponto de vista do reconhecimento. A convenção adotada para sanar em definitivo o problema foi a mesma que é usada no processador de textos LaTeX, largamente difundido na comunidade acadêmica: expressões matemáticas devem ser envolvidas no texto por um par de cifrões (`$ ... $`); além disso, os identificadores que fazem menção a funções ou a primitivas de exibição devem ser introduzidos por um mesmo caractere – a contrabarra: \.

Mantivemos da versão anterior um repertório próprio em português para os nomes das funções e das primitivas de exibição mais utilizadas (a listagem destes nomes encontra-se no Apêndice A). A conversão para os nomes efetivamente aceitos pelo Maxima e pelo filtro que precede a renderização final no navegador é feita de maneira transparente para o usuário. Assim, o usuário não precisa conhecer os nomes das funções do Maxima e das primitivas de exibição em inglês para ser capaz de redigir seus textos. Só os principais comandos do Maxima (são centenas!) integram este pequeno dialeto que procuramos disponibilizar. Este subconjunto pode ser estendido, na presente implementação, apenas pelo administrador da plataforma, incorporando novas funções e primitivas ao repertório; no futuro, pretendemos que esta extensão possa ocorrer dinamicamente, por iniciativa do instrutor de um curso, por exemplo.

Na transição da versão preliminar para a versão atual do MathMoodle, o LMS Moodle sofreu, por parte de seus desenvolvedores, uma considerável reestruturação. Em vez do antigo HTMLArea, ao qual o nosso plug-in MathWriting se acoplava, um editor de textos bem mais completo e amigável passou a ser utilizado como padrão: o TinyMCE. Este editor, já distribuído nas novas versões oficiais do Moodle, possui desenvolvimento e manutenção independentes do LMS. Ele é invocado em todo módulo em que a edição de textos se faça necessária. Desta forma, acoplando o MathWriting ao TinyMCE, estendemos a capacidade de editar e visualizar textos com conteúdo matemático à maioria dos módulos do Moodle.

Na implementação que realizamos, modificamos o editor de modo a integrá-lo com o MathWriting. Para isto, a tela do TinyMCE foi dividida em duas áreas: a superior contendo o texto sendo editado e a inferior contendo a visualização resultante do processamento, conforme é mostrado na Figura 2.1. Um botão específico foi acrescentado ao painel do editor para comandar a execução do MathWriting e a conseqüente exibição do texto editado.

Outra necessidade que notamos durante a reestruturação do MathMoodle foi a de oferecer ao usuário mais experiente e acostumado a programar a possibilidade de delegar ao Maxima uma sequência arbitrariamente longa de comandos que não produzem de imediato saída visualizável. Nestes trechos, aqui denominados silenciosos,

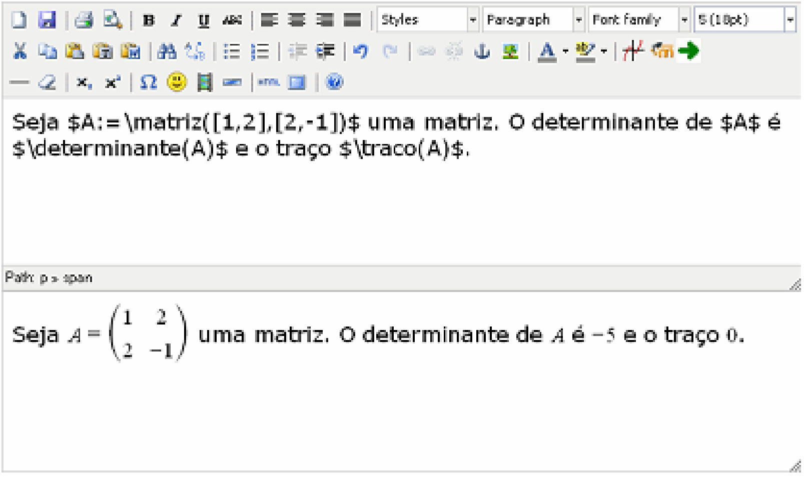


Figura 2.1: A aparência modificada do editor TinyMCE

variáveis podem ser alteradas, em função do processamento designado pelos comandos do usuário. Entretanto, nenhum dos valores calculados é imediatamente exibido. Posteriormente no texto, através do operador `@()`, os valores destas variáveis podem ser recuperados e exibidos onde forem necessários. A esta nova funcionalidade denominamos *modo silencioso de processamento*, ilustrado na Figura 2.2.

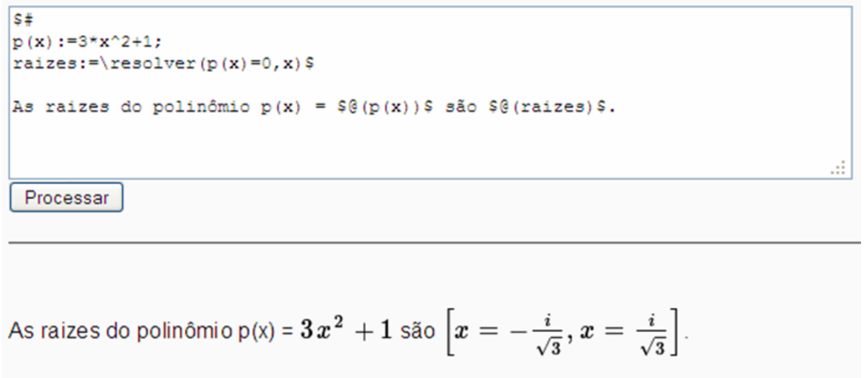


Figura 2.2: Modo silencioso de processamento

A sequência de comandos é delimitada por um par de cifrões, exatamente como qualquer expressão matemática. Só que, neste caso, o primeiro cifrão é seguido pelo caractere sustenido (#), indicando que o trecho compreendido daquele ponto até o cifrão final deverá ser processado pelo Maxima, mas que a resposta emitida por este não deverá ser inserida no texto, como aconteceria a uma expressão matemática normal.

O trecho delimitado por `$# ... $` na Figura 2.3 armazena na variável `A` uma matriz 3×3, na variável `det` o determinante e na variável `tra` o traço de `A`. Nenhuma saída visualizável é produzida em virtude deste trecho; somente em pontos posteriores do texto, através das expressões `@(A)`, `@(det)` e `@(tra)`, os valores destas variáveis são recuperados e efetivamente exibidos.

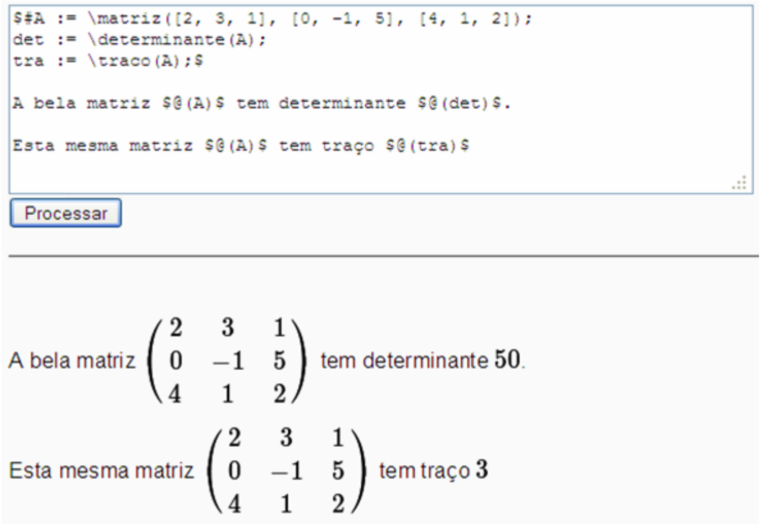


Figura 2.3: Interação em modo silencioso

No modo silencioso, valem atribuições a variáveis e também todos os nomes de funções do repertório em português, que são convertidos para os nomes de funções conhecidos pelo Maxima.

Por fim, uma última funcionalidade, derivada do modo silencioso, nos ocorreu para esta nova versão do MathMoodle: oferecer ao usuário fluente a possibilidade de emitir comandos diretamente na sintaxe nativa do Maxima, sem estar restrito ao repertório de nomes traduzidos que fixamos. À semelhança do modo silencioso, este novo modo de interação, que denominamos *literal*, não emite resultado visível na exibição do documento: a ideia é que o usuário modifique valores de variáveis e os recupere posteriormente no texto, através do mesmo operador `@()`.

Um trecho em modo literal é delimitado por `$! ... $` (`!` é utilizado em vez de `#`, para distinguir do modo silencioso). No exemplo da Figura 2.4, o trecho entre `$! ... $` a ser processado pelo Maxima encarrega-se de gerar aleatoriamente uma matriz `A`, com entradas inteiras, que possui autovalores `a` e `b`, também inteiros e gerados aleatoriamente.

Vale ressaltar que, no modo literal, o Maxima pode ser utilizado em toda a sua potencialidade, já que o usuário especifica os comandos diretamente na sintaxe nativa e nenhum processo de tradução é efetuado. No exemplo da Figura 2.4, até um comando iterativo de programação é utilizado (`while`). É claro que não visamos, com esta funcionalidade, o usuário comum, que não domina a complexa linguagem aceita pelo Maxima; o objetivo foi contemplar os especialistas, que ficariam injustamente restritos a um repertório limitado, caso o modo literal não existisse.

Do ponto de vista da implementação, não vamos abordar nesta dissertação os detalhes técnicos envolvidos, que seriam: a reprogramação do analisador sintático,

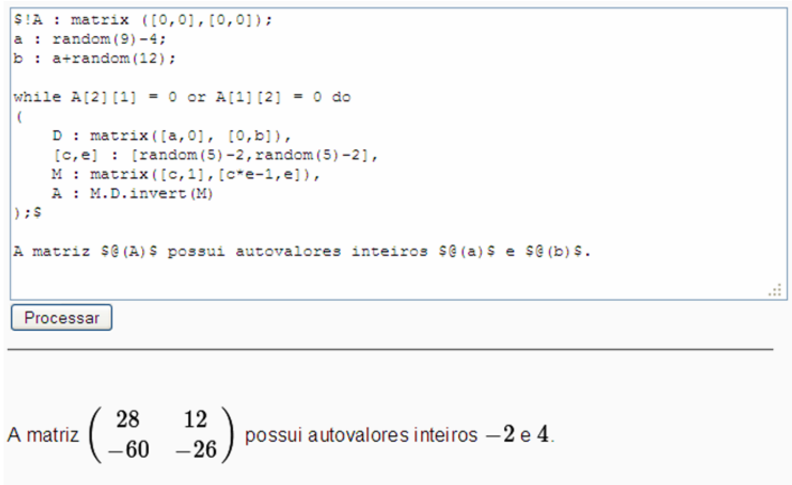


Figura 2.4: Interação em modo literal

a gerência unificada da tabela de símbolos contendo nomes de funções, o processo de tradução de nomes e a comunicação interprocessual entre o MathWriting e o Maxima. Enfatizamos apenas que as alterações produzidas no Moodle foram mínimas e pontualmente localizadas e que, para isto, contribuiu muito a arquitetura flexível segundo a qual ele foi concebido.

Destacamos, entretanto, que uma mudança significativa se produziu no que diz respeito ao processamento das primitivas de exibição presentes nas expressões matemáticas. A versão anterior do MathMoodle utilizava um precário filtro conversor, denominado ASCIIMathML (JIPSEN, 2007, [8]), que era responsável por traduzir os nomes das primitivas de exibição em etiquetas MathML, interpretadas pelo navegador no processo de renderização. Este filtro, além de não ser mais atualizado ou melhorado há alguns anos, possuía severas limitações quanto às primitivas aceitas e exigia a instalação de fontes específicas no navegador, para que o processo de renderização se efetuasse corretamente. Isto limitava o uso do MathMoodle a certos navegadores, impossibilitando-o em outros. Do ponto de vista da portabilidade, executar em qualquer navegador é uma enorme vantagem, uma vez que poucos são os usuários capacitados a instalar fontes e plug-ins.

A versão presente do MathMoodle utiliza um novo filtro conversor, denominado MathJax[1], que aceita como entrada primitivas de exibição no estilo LaTeX e é capaz de produzir saída para três modalidades distintas de renderização: as velhas conhecidas etiquetas MathML, as primitivas de estilo CSS ou o código vetorial SVG. A escolha é feita automaticamente, dependendo das possibilidades do navegador utilizado, de forma totalmente transparente para o usuário e sem a necessidade de instalações adicionais. A difusão deste filtro de conversão na internet simplificou

[1] `www.mathjax.org`

sobremaneira nosso trabalho, principalmente porque a sintaxe que havíamos adotado para as primitivas já se inspirava em LaTeX, que é exatamente o esperado pelo MathJax.

2.3 Os Scripts e o Módulo Scripts

A introdução dos modos silencioso e literal, este último permitindo ao usuário interagir com o Maxima diretamente em sua sintaxe nativa, teve um impacto inovador no uso da plataforma. Em pouco tempo, percebemos o surgimento de um modelo de redação (um *template*, para usar o termo em inglês), segundo o qual um texto matemático deve constituir-se da intercalação de seções, que podem ser de dois tipos:

- **Seções algorítmicas**: compreendendo trechos matemáticos em modo silencioso (`$# ... $`) ou literal (`$ ... $`). Nestas seções, algum processamento matemático é descrito algorítmicamente através de atribuições e comandos do Maxima e, nele, variáveis chaves são alteradas de forma a armazenar valores que estarão disponíveis às seções subsequentes.
- **Seções de texto propriamente dito**: em que os valores das variáveis que foram computados nas seções algorítmicas precedentes são resgatados através de um operador específico: `@()`, sempre que se façam necessários.

A um documento de texto redigido nestes moldes denominamos *script*. A Figura 2.4 nos apresentou um exemplo com apenas duas seções: a primeira algorítmica e a segunda textual.

Os benefícios de escrever os textos matemáticos seguindo este modelo foram de imediato percebidos pela equipe que nos auxiliava a depurar a ferramenta: além da clareza inerente à estrutura, os textos assim redigidos tornavam-se autodocumentados, facilmente alteráveis e adaptáveis a outras finalidades.

A partir destas constatações, surgiu-nos a ideia de criar um novo módulo no Moodle, que batizamos de *Módulo Scripts*. Ele pode ser incorporado como tópico em qualquer parte de um curso. Neste módulo, o instrutor disponibiliza os scripts que julgar convenientes aos demais participantes do curso, designando-os por títulos. Os títulos integram um menu de seleção e, clicando sobre um deles, provoca-se a transferência do conteúdo do respectivo script para uma janela do editor TinyMCE, onde ele pode ser processado e o resultado do processamento ser exibido.

Além de comandar o processamento e a exibição, o usuário pode, evidentemente, alterar o script antes de mandar processá-lo, quer modificando seções algorítmicas, quer alterando seções de texto.

É exatamente nesta possibilidade que reside um grande potencial didático desta integração que estamos descrevendo: os participantes do curso podem, através de mudanças em um script disponibilizado pelo instrutor, explorar uma ampla gama de variações de cenários sobre o conteúdo de que trata o script. Estas variações podem, inclusive, ser sugeridas pelo autor no próprio corpo do script, num processo de estudo dirigido que estimula o participante motivado a explorar outros rumos relevantes ao entendimento do tema. Deste modo, o script representa um roteiro predefinido e planejado para explorar ou enfatizar determinados aspectos de aprendizagem. O instrutor pode, por exemplo, criar uma lista de exercícios dinâmica em que, a partir de alguns exemplos iniciais, os alunos podem propor e resolver exercícios correlatos.

O próprio instrutor, dispondo dos aparatos necessários em sala de aula (basicamente um computador ligado à internet e um projetor), pode beneficiar-se do uso de scripts durante as aulas, para ilustrar de diferentes maneiras o assunto em pauta. Variando os dados de um script, obtêm-se cenários diversos, que podem ser analisados e discutidos durante a aula, enriquecendo-a do ponto de vista didático.

Para os alunos, há ainda um ganho colateral que seria mais difícil de obter se o Maxima fosse usado isoladamente, sem o MathMoodle: a possibilidade de ir assimilando gradualmente, através de um processo de colagem e adaptação de scripts, a linguagem nativa do Maxima, compreendendo sua sintaxe e explorando recursos que, de outra forma, exigiriam a leitura de um extenso manual (MAXIMA, 2012, [10]).

Uma das vantagens principais desta ferramenta que propomos é a singularidade desta integração entre um LMS e um CAS: por um lado, agregamos ao CAS uma interface amigável de uso e, por outro, acrescentamos ao LMS a capacidade de processar e exibir, com qualidade, conteúdo matemático.

Este ganho é potencializado na medida em que o uso remoto via internet e as possibilidades de replicar este modelo em disciplinas permitem o acesso em larga escala por parte da comunidade a uma ferramenta para comunicação efetiva e processamento de conteúdos matemáticos.

No Capítulo 3 e nas conclusões teremos a oportunidade de discutir o quanto os scripts se destacaram como valiosos instrumentos de complementação didática no experimento de validação que realizamos.

2.4 A Tecnologia de Redação Matemática

Concluída a reimplementação do MathMoodle, detivemo-nos em uma reflexão do ponto de vista estrutural acerca da ferramenta que havíamos produzido, com o intuito de perceber quais funcionalidades estavam concentradas em quais módulos da ferramenta.

Em geral, o processo de análise empregado em sistemas de computação prega exatamente o caminho inverso do que fizemos: primeiro identificam-se as necessidades do usuário, as funcionalidades requeridas; depois, concebe-se a estrutura em módulos, suas responsabilidades e sua interconexão; somente em uma etapa posterior deve partir-se para a implementação, que eventualmente realimenta a análise com os inevitáveis imprevistos que ocorrem.

Em nosso caso, sendo o MathMoodle um objeto em pesquisa, os fatos não se deram nesta ordem. Possuíamos uma ferramenta experimental, rudimentar, que foi evoluindo ao longo do tempo em virtude de observações e sugestões coletadas em alguns experimentos. A ferramenta foi reimplementada integralmente, conforme já descrevemos. Somente após a sua estabilização e algum tempo de uso, nos foi possível observar o que havíamos produzido sob uma ótica analítica, buscando pela essência funcional e modular, distante dos detalhes que dizem respeito à implementação.

Esta *análise essencial* posterior à implementação possibilitou-nos identificar com clareza três componentes na arquitetura proposta para o MathMoodle, que serão aqui denominados *agentes*:

1. O Agente de Captação

   Este agente é responsável pela análise sintática do texto fornecido pelo usuário. Desta análise, resulta a identificação, a extração e o pré-processamento de expressões matemáticas. Este pré-processamento consiste em realizar a tradução dos nomes de funções e primitivas previstos no dialeto em português para os nomes efetivamente reconhecidos pelos outros agentes.

   Após a análise sintática, o Agente de Captação invoca o Agente de Processamento, descrito no item seguinte, enviando-lhe os trechos matemáticos já convertidos para sua linguagem nativa e esperando como resposta os resultados do processamento.

   Obtidas estas respostas, o Agente de Captação as insere no texto no exato lugar onde havia as expressões matemáticas originais, atentando para o processamento especial dos modos silencioso e literal.

2. O Agente de Processamento

   Este agente é capaz de processar numerica e algebricamente expressões matemáticas complexas, envolvendo variáveis, funções e até mesmo comandos de programação que permitem a elaboração de algoritmos bastante sofisticados.

Seu papel na arquitetura é reagir a solicitações do Agente de Captação, efetuando o processamento matemático e retornando as respostas requeridas.

Evidentemente há um protocolo de comunicação entre estes dois primeiros agentes, que, do ponto de vista desta análise, não é relevante.

3. O Agente de Exibição

   Este agente tem por finalidade produzir a exibição final do texto, renderizando as expressões matemáticas em um formato inteligível, semelhante em qualidade ao que se encontra em livros impressos de matemática.

   É claro que a qualidade da renderização dependerá do meio disponível para visualização escolhido, que pode ser uma folha de papel onde o texto será impresso ou o interior da janela em de um navegador, executando possivelmente em um computador de mesa ou portátil, como notebooks, tablets, telefones e outros.

Na Figura 2.5 está ilustrada a interconexão entre estes agentes que acabamos de detalhar.

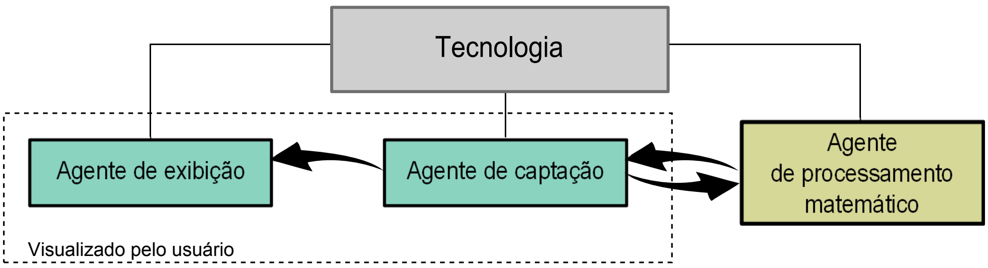


Figura 2.5: A Tecnologia de Redação e seus Componentes

Na arquitetura do MathMoodle, estes componentes estão bem evidentes: o Agente de Captação é representado pelo plug-in MathWriting, o Agente de Processamento é o próprio CAS Maxima e o Agente de Exibição é constituído pelo filtro conversor MathJax e pelo renderizador matemático do navegador Web.

Entretanto, este não é o único cenário possível em que esta estrutura trimodular se materializa, bem como as escolhas feitas para a implementação dos agentes não estão limitadas às que fizemos.

No experimento de validação que realizamos, a ser descrito no Capítulo 3, ficou evidente, durante o acompanhamento das aulas, que, devido às restrições de resolução dos projetores disponíveis, a área da janela do navegador que restava para exibição simultanea de um script juntamente com o resultado de seu processamento era insuficiente, tendo em conta que o próprio Moodle ocupa parte considerável deste espaço.

A solução foi criar uma nova funcionalidade para o editor TinyMCE, que permitisse a visualização simultânea de um script e do resultado de seu processamento lado a lado na janela do navegador. Um botão específico foi acrescentado na barra de ferramenta do editor para ativar este modo de exibição. Esta funcionalidade potencializa a exploração didática da tecnologia, uma vez que toda modificação no script implica a imediata exibição em uma área adjunta, como pode ser visto na Figura 2.6.

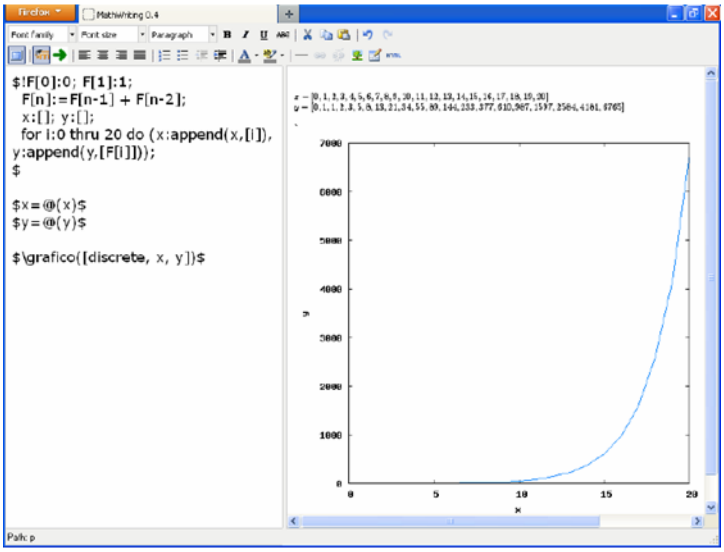


Figura 2.6: Script e resultado do seu processamento lado a lado

Testes com esta nova funcionalidade apontaram que sua utilização isolada do MathMoodle melhorava a clareza com que a exposição era projetada durante as aulas. Por esta razão, criamos uma ferramenta complementar que não utiliza o Moodle, mas apenas o editor TinyMCE executando independentemente. A mesma estrutura modular de três agentes que especificamos nesta análise foi utilizada na concepção desta nova ferramenta, por nós batizada como *mwEdPlayer*, revelando-se mais adequada ao uso em sala de aula do que o próprio MathMoodle.

Em um contexto totalmente diverso, esta tecnologia de três agentes está sendo utilizada na implementação do sistema AtenaME (VERNET, 2012, [15]) desenvolvido também no LIMC/UFRJ, cujo objetivo é gerar e corrigir automaticamente testes de múltipla escolha (VERNET, 2012, [15]; GUIMARÃES *et al*, 2012, [5]).

Neste sistema, é possível, por meio da mesma sintaxe do modo literal utilizada em scripts, especificar enunciados de questões que contenham elementos dependentes de algum processamento algorítmico, de modo que um lote de testes de múltipla escolha possua questões com enunciados equivalentes, porém com dados distintos gerados aleatoriamente.

No AtenaME, uma versão simplificada do Agente de Captação foi embutida

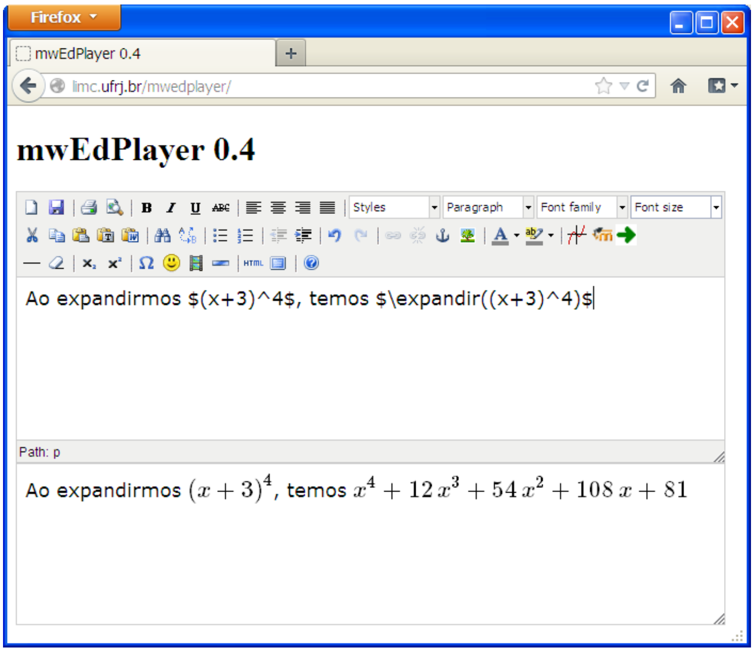


Figura 2.7: Ferramenta mwEdPlayer

no programa de geração de testes. Os enunciados das questões de um teste são especificados em um arquivo, cuja sintaxe é descrita em (VERNET, 2012, [15]). Nesta implementação do Agente de Captação, apenas o modo literal de interação é suportado, de maneira que cada questão integrante do teste possa ter a estrutura de um script de duas seções: um algoritmo onde dados são computados e o texto propriamente dito do enunciado, em que os valores destes dados são recuperados através do mesmo operador `@()`.

O Agente de Processamento, evidentemente, é o próprio Maxima e o Agente de Exibição é o processador de textos LaTeX, juntamente com os programas acessórios que permitem renderizar os testes gerados no formato PDF, para posterior impressão e aplicação.

Observamos, então, a concretização de uma mesma estrutura modular, composta dos três agentes citados, em contextos computacionais independentes. Este fato nos motiva a caracterizar esta estrutura tríplice como uma *tecnologia para redação de textos matemáticos*, que pode se materializar em realidades computacionais diversas, conforme está sendo por nós constatado em nosso dia a dia de desenvolvimento no LIMC/UFRJ.

Capítulo 3

O Experimento de Validação

Conforme descrevemos no capítulo anterior, a evolução do MathMoodle passou por diversas etapas, algumas delas concomitantes ao experimento que iremos relatar.

O objetivo foi o de validar as ferramentas que havíamos desenvolvido como instrumentos didáticos coadjuvantes no ensino de uma disciplina regular de matemática, em nível de graduação. Este processo de validação procurou verificar se o uso planejado das ferramentas em sala de aula e em atividades extraclasse poderia contribuir positivamente para um melhor andamento do curso, para um melhor aproveitamento por parte dos alunos e para atenuar certas dificuldades que, segundo relatos de alguns professores, parecem ocorrer com frequência no ensino da disciplina escolhida.

Neste capítulo, portanto, resumimos nossas observações acerca desta experiência de validação, que utilizou como campo a disciplina de Álgebra Linear II, oferecida por dois períodos letivos sucessivos a alunos das Engenharias.

De início, apontamos algumas dificuldades relatadas por professores com quem conversamos, bastante experientes no ensino desta disciplina. Elas serviram de motivação primeiramente para a escolha da disciplina e depois para o planejamento de como o uso das ferramentas poderia se dar de modo a interferir beneficamente nas diversas atividades do curso, numa tentativa de amenizar algumas destas dificuldades levantadas.

Na seção seguinte, descrevemos uma etapa prévia, que consistiu no planejamento do curso junto ao professor, orientador desta dissertação, que ministraria a disciplina.

Em continuação, os principais pontos em que se deu a efetiva utilização do MathMoodle no curso são descritos. Procuramos mostrar, em cada um dos quatro tópicos escolhidos, por que o uso da ferramenta foi planejado para o referido tópico, como ele ocorreu, as modificações que este uso causou na abordagem e os benefícios didáticos foram observados.

Na seção final, concluímos com algumas apreciações sobre a validação, que atestam positivamente o que pretendíamos validar.

3.1 A Disciplina de Álgebra Linear para Graduação

Uma etapa preliminar à validação que iremos descrever consistiu em identificar, em conversas que tivemos com alguns professores que ministram regularmente a disciplina de Álgebra Linear II para turmas de graduação, os principais desafios que eles costumam enfrentar. Procuraremos, nesta seção, resumir o que nos foi relatado.

Esta disciplina é cursada a cada semestre, na UFRJ, por mais de 800 alunos oriundos de diversas áreas, normalmente no final do primeiro ano de curso (segundo período de aulas). Ela está incluída nos currículos de todas as carreiras técnicas: engenharia, física, química, arquitetura, ciência da computação e, claro, matemática.

Um primeiro desafio que alguns professores relatam é o de conseguir adaptar o curso coerentemente a fim de atender a necessidades específicas de cada área. A dificuldade reside em como graduar adequadamente esta adaptação sem prejuízo dos aspectos conceitual e operacional, ambos inerentes a esta disciplina e igualmente importantes.

A ementa usual para um curso de quatro meses (um período letivo) parece bastante ambiciosa: Matrizes e Determinantes, Sistemas de Equações Lineares, Espaços Vetoriais, Dependência e Independência Linear, Base e Dimensão, Transformações Lineares, Operadores Lineares, Espaços com Produto Interno, Autovalores e Autovetores.

Alguns destes tópicos possuem desenvolvimentos que requisitam cálculos e esta peculiaridade acaba por desviar a abordagem dos assuntos para um enfoque mais operacional do que os professores gostariam que fosse, principalmente os matemáticos, restando pouco tempo para uma maior atenção aos conceitos. Sem o devido cuidado, o curso pode acabar se degenerando em uma coleção de receitas de cálculo, onde os conceitos teóricos são menosprezados. Isto pode ocorrer, em especial, em cursos de Álgebra Linear oferecidos às engenharias, nos quais uma abordagem mais aplicada é esperada. Entretanto, uma abordagem mais aplicada não deveria ser confundida com uma abordagem mecanizada. Conforme assinala (WAWRO *et al*, 2011, [16]), a ênfase nas aplicações em detrimento da parte conceitual acarreta percepções distorcidas, como "...há pouco tempo para espaços vetoriais e transformações lineares..."; ou ainda, a omissão de um conceito tão fundamental quanto o de isomorfismo acaba produzindo nos alunos a falsa idéia de que, mesmo se $k < n$, $\mathbb{R}^k$ é um subespaço de $\mathbb{R}^n$".

Por outro lado, a Álgebra Linear pode também ser ensinada de maneira formal e abstrata, onde o rigor algébrico é privilegiado e quase nada do aspecto operacional é abordado. Mesmo neste caso, tudo pode se desenvolver de forma totalmente mecanizada, já que um uso cego e puramente sintático da linguagem algébrica pode levar os alunos à mera imitação de padrões, sem a verdadeira compreensão do que estão fazendo. Isto pode ocorrer em cursos de Álgebra Linear oferecidos aos estudantes de matemática.

Pelo que recolhemos de nossas conversas com os professores, não lhes parece trivial encontrar o devido equilíbrio entre teoria e aplicações, atendendo a necessidades específicas de cada área a que o curso é oferecido.

Um segundo desafio apontado é a escolha da bibliografia. Existem excelentes livros sobre Álgebra Linear mais avançada: (AXLER, 1997, [1]) e (HALMOS, 1974, [6]) são conhecidos exemplos. No entanto, nenhum deles é adequado a uma primeira abordagem. Por outro lado, embora exista extensa literatura dedicada a cursos preliminares, a maioria dos livros tem dificuldade em proporcionar o equilíbrio adequado entre conceitos e aplicações. Conceitos relacionados à geometria em Álgebra Linear são algumas vezes introduzidos sem que se apresente a correspondente interpretação geométrica, que, se não obrigatória, seria pelo menos desejável para complementar e facilitar a compreensão. Além disso, o salto de velhos hábitos da álgebra escolar para o raciocínio mais geral e abstrato, e as dificuldades que os alunos encontram ao longo do caminho, são difíceis de serem resolvidas sem uma rica galeria de exemplos aos quais recorrer. Adotar um único livro como referência bibliográfica é insuficiente para cobrir todos os temas com as qualidades desejadas para um público específico; adotar vários livros faz surgir novos problemas devidos à disparidade de notações e definições.

Algumas das abordagens "consagradas" no ensino de Álgebra Linear são verdadeiramente perniciosas no nível conceitual. Para citar um exemplo, um tema presente em todos os livros introdutórios é o reconhecimento da (in)dependência linear entre um conjunto de vetores dados. Isto é realizado organizando os vetores em colunas de uma matriz, que sofre uma transformação para algum tipo de forma escalonada. Nesta forma, surgem zeros em posições da matriz que permitem concluir se os vetores são ou não são linearmente dependentes.

A maneira como este procedimento é tratado produz resultados decepcionantes: os alunos assimilam o "algoritmo de escalonamento" como uma receita milagrosa de cálculo, totalmente alienada de qualquer justificativa matemática. Diante de uma matriz, o impulso de escalonar torna-se incontrolável da parte do aluno, não importa qual seja a questão que está sendo discutida.

Outro tópico que é comumente mecanizado nos cursos é o estudo dos autovalores e autovetores de um operador linear. Em alguns textos, etapas essenciais de motivação, definição e caracterização são praticamente puladas, sendo a questão abruptamente reduzida à determinação das raízes do polinômio característico, o que só é viável em dimensões baixas. Concentrar-se em dimensão 2 ou 3 é razoável para um curso introdutório, mas não se pode perder de vista que, bem antes de se chegar ao "cálculo" dos autovalores, há perguntas chaves que motivam o tema e que, muitas vezes, são omitidas: Existem direções que permanecem invariantes sob o operador em questão? Quais são elas? Quantas podem ser? Qual sua importância? A partir destas indagações em torno do conceito de invariância é que se deveria passar ao desenvolvimento propriamente dito do tópico. Neste ponto, boa parte da literatura introdutória disponível mistura de forma confusa definição, caracterizações e o cálculo dos autovalores, prosseguindo com aplicações variadas e díspares, sem a devida ênfase na questão da invariância – por exemplo (GONÇALVES & SOUZA, 1977, [4]), (STEINBRUCH & WINTERLE, 1990, [14]). Novamente mais um tópico se operacionaliza mecanicamente com prejuízo do conceito que o motiva.

Ficou claro para nós que esta disciplina oferecia um terreno propício ao experimento de validação que pretendíamos: o uso das ferramentas que desenvolvemos poderia ser proveitoso tanto para o professor, no seu esforço de reaproximar a abordagem de alguns tópicos mais "operacionais" de sua essência conceitual, quanto aos alunos, que se beneficiariam com uma aula mais dinâmica, mais rica em exemplos e ilustrações, além de poderem também contar com os recursos didáticos disponibilizados para atividades externas à sala de aula.

3.2 Planejando o Experimento

Levando em conta os aspectos apontados na seção anterior, concluímos que seria oportuna a utilização das ferramentas que desenvolvemos tanto em sala de aula quanto em atividades extraclasse. No primeiro caso, a ferramenta principal seria o mwEdPlayer, que possibilita ao professor processar scripts durante as aulas, alterando-os quando necessário a fim de produzir as variações pertinentes à ilustração do tópico em pauta. No caso de atividades extraclasse, o Módulo Scripts, que acrescentamos ao Moodle, permite ao professor deixar disponíveis os scripts exibidos em aula, bem como os scripts que constituíam roteiros de estudo dirigido, a serem trabalhados fora do horário de aula.

Em se tratando de um experimento que interferiria tanto na dinâmica das aulas quanto nas atividades extraclasse, seus efeitos não eram previsíveis e seu sucesso não era garantido. Desta forma, nas reuniões preliminares que tivemos com o professor, ficou decidido que o uso das ferramentas seria restrito a algumas aulas e tópicos específicos.

Os tópicos escolhidos para serem tratados de forma diferenciada foram justamente o escalonamento e estudo dos autovalores e autovetores, pelos motivos que já apresentamos na seção anterior: nestes pontos, era preciso trabalhar o aspecto operacional, realizando os cálculos necessários, mas também dando ênfase aos conceitos teóricos envolvidos, de modo que o conteúdo não fosse assimilado pelos alunos como um conjunto de "receitas de cálculo".

Um outro ponto importante que ficou acertado logo de início foi o envolvimento mais próximo e com maior comprometimento de uma equipe selecionada de monitores, ex-alunos recentes da disciplina, que estariam dedicados a examinar regularmente as postagens feitas pelos alunos no MathMoodle, contribuindo com sugestões para a resolução de exercícios, respondendo a dúvidas acerca da matéria e do uso da plataforma. Reuniões regulares seriam realizadas entre o professor e esta equipe, com nossa participação, nas quais as orientações seriam passadas e eventuais correções de rumo seriam feitas.

Nos tópicos escolhidos pelo professor, seria necessária a nossa colaboração no sentido de redigir e depurar os scripts, tarefa esta que foi empreendida no decorrer do curso, sob a orientação teórica do professor, à medida que a disciplina avançava.

3.3 A Intervenção das Ferramentas no Curso

Nesta seção, relatamos algumas das contribuições do uso das ferramentas MathMoodle e mwEdPlayer durante os dois cursos de Álgebra Linear II que foram observados durante a validação.

3.3.1 Unificando Notações e Definições

Um dos problemas já citados é o da multiplicidade de notações e a discrepância entre algumas definições que se encontram nos livros recomendados aos alunos como referências bibliográficas para o curso. Com a queda na qualidade do ensino secundário presenciada nas últimas décadas, não se pode mais esperar de um aluno, que chega à universidade despreparado e com pouco ou nenhum autodidatismo, que apresente a maturidade mínima que permitiria conciliar, por conta própria, essas diferenças de notações e definições entre fontes diversas.

O Moodle originalmente oferece um recurso às vezes subestimado: o glossário. Trata-se de uma coleção de entradas com textos curtos pode ser organizada em ordem alfabética, juntamente com o texto associado contendo significados ou explicações. Com o MathMoodle, é possível uma entrada de glossário apresentar conteúdo matemático, que pode a qualquer momento ser objeto de alterações e melhorias.

Este recurso se mostra muito útil para estabelecer uma terminologia padrão para o curso, incluindo as definições dos principais termos matemáticos, que podem variar de um livro para outro. Assim, um glossário solidamente construído, com a cooperação entre professor e alunos ao longo do período, é uma maneira bastante adequada de unificar a linguagem e a notação utilizadas na disciplina. Destacamos, na Figura 3.1 a seguir, um exemplo de entrada de glossário introduzido pelo professor.

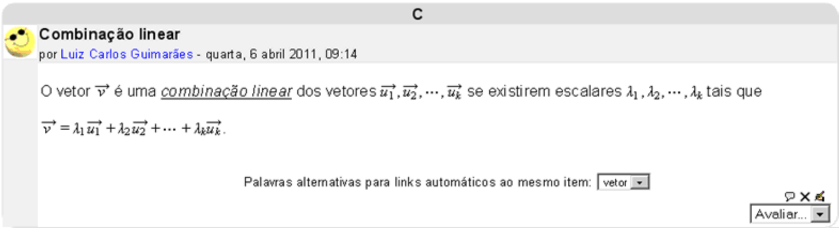


Figura 3.1: Entrada de glossário definindo Combinação Linear

O glossário foi utilizado no curso de maneira totalmente colaborativa. A utilização consistia em o professor propor o título e preencher total ou parcialmente o conteúdo da respectiva entrada, dependendo do grau de dificuldade envolvido na conceituação. Era solicitado aos alunos que, fazendo acesso remoto ao site do MathMoodle, inspecionassem, completassem e alterassem a entrada da maneira como julgassem necessário. A equipe de monitores auxiliava no processo, assinalando os erros cometidos ou, muitas vezes, cometendo erros piores, o que permitiu ao professor retrabalhar com eles muitos conceitos que haviam ficado mal compreendidos em sua passagem prévia pela cadeira de Álgebra Linear.

Coube a nós, juntamente com o professor, identificar as dificuldades que ficavam aparentes por parte dos alunos, de modo a serem trabalhadas nas aulas subsequentes.

3.3.2 Gerando Exemplos com Características Conhecidas

Uma aula tradicional de algumas disciplinas de matemática, ao quadro-negro, exige uma preparação prévia, por parte do professor, de exemplos numéricos. No caso da Álgebra Linear, estes exemplos envolvem principalmente matrizes, cálculo de determinantes, soluções de sistemas lineares, dentre outros.

A menos que o professor esteja muito motivado em criar suas próprias alternativas, o que ocorre com mais frequência é que estes exemplos sejam extraídos diretamente das fontes bibliográficas, e que sejam, por vezes, tomados sem o devido critério. Em consequência, eles podem não contribuir significativamente para uma compreensão mais ampla por parte dos alunos do tópico sendo ensinado, por não possuírem certas características que possibilitem a ilustração de aspectos de fato relevantes dentro do assunto sendo ensinado.

A preparação de exemplos relevantes do ponto de vista didático exige, por parte do professor, um certo amadurecimento no ensino da disciplina, de modo a ter clareza das diversas ramificações que um determinado tópico pode tomar e poder explorá-las consistentemente. Isto, é claro, vale não apenas para Álgebra Linear.

Vamos explicitar aqui um caso mais simples em que o uso de scripts no MathMoodle facilitou a geração de exemplos que tivessem certas características conhecidas, de modo a serem usados em sala em aula.

No estudo dos autovalores e autovetores, um cuidado especial é necessário na escolha das matrizes que servirão de base aos exemplos. Ainda que uma matriz tenha entradas inteiras, é possível que ela não possua autovalores reais. Muitas vezes, é isto o que se quer ilustrar; mas, em outras tantas, o que o professor deseja é que as contas envolvidas sejam simples o suficiente para poderem ser trabalhadas no quadro-negro sem consumir tempo excessivo de aula e, de preferência, sem incorrer em erros desnecessários de cálculo. Assim, numa explanação inicial sobre este tema, pode ser muito oportuna a capacidade de gerar matrizes "bem comportadas", isto é, com entradas representadas por números inteiros pequenos, de modo que os autovalores sejam também inteiros e pequenos, fáceis de calcular.

Em nosso experimento, o professor nos solicitou que construíssemos um script de geração de matrizes 2×2 que tivessem estas características. A ideia proposta por ele consistia em pensar o problema ao inverso: em vez de dada a matriz, obter os autovalores, deveríamos pré-estabelecer dois inteiros pequenos e obter uma matriz 2×2 com entradas inteiras também pequenas e que tivesse os números fixados como autovalores.

A solução consiste em partir de uma matriz diagonal D com os autovalores λ_1 e λ_2 conhecidos ocupando a diagonal principal. É claro que esta matriz satisfaz os requisitos desejados, mas não é muito interessante produzir exemplos apenas com matrizes diagonais. Dado que matrizes quadradas similares [1] possuem os mesmos autovalores, qualquer matriz similar a D terá λ_1 e λ_2 como autovalores. Assim, a matriz desejada pode ser obtida multiplicando D à esquerda por uma matriz M invertível e, à direita, pela inversa de M. Entretanto, M e sua inversa deverão também ter entradas inteiras, o que pode ser conseguido fixando que o determinante de M valha 1. O processo é descrito algebricamente a seguir.

Sejam λ_1 e λ_2 os autovalores dados e seja D a matriz diagonal com estes autovalores:

[1] A e B são similares quando existir uma matriz invertível M tal que $B = M \cdot A \cdot M^{-1}$.

$$D = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix}$$

A matriz E, similar a D, que procuramos é dada por $E = M \cdot D \cdot M^{-1}$, onde $M = \begin{bmatrix} a & b \\ c & d \end{bmatrix}$ é invertível e $M^{-1} = \dfrac{1}{ad - bc} \begin{bmatrix} d & -b \\ -c & a \end{bmatrix}$. Logo,

$$E = \frac{1}{ad - bc} \begin{bmatrix} a & b \\ c & d \end{bmatrix} \cdot \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} \cdot \begin{bmatrix} d & -b \\ -c & a \end{bmatrix}.$$

Assumindo que as entradas de M sejam inteiras, para que M seja invertível e sua inversa também tenha entradas inteiras, basta impormos que o determinante $ad - cb$ seja igual a 1. Para diminuir o número de parâmetros a escolher, tomamos também $d = 1$. Assim, $a = 1 + cb$. Logo:

$$E = \begin{bmatrix} 1 + cb & b \\ c & 1 \end{bmatrix} \cdot \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} \cdot \begin{bmatrix} 1 & -b \\ -c & 1 + cb \end{bmatrix}.$$

Portanto, pré-estabelecendo os autovalores λ_1 e λ_2 e atribuindo valores inteiros a b e c, obteremos a matriz desejada.

O script correspondente já foi apresentado na Figura 2.4 do Capítulo 2, como exemplo de interação em modo literal. Repetimos a figura aqui apenas para que o leitor não precise retroceder muitas páginas:

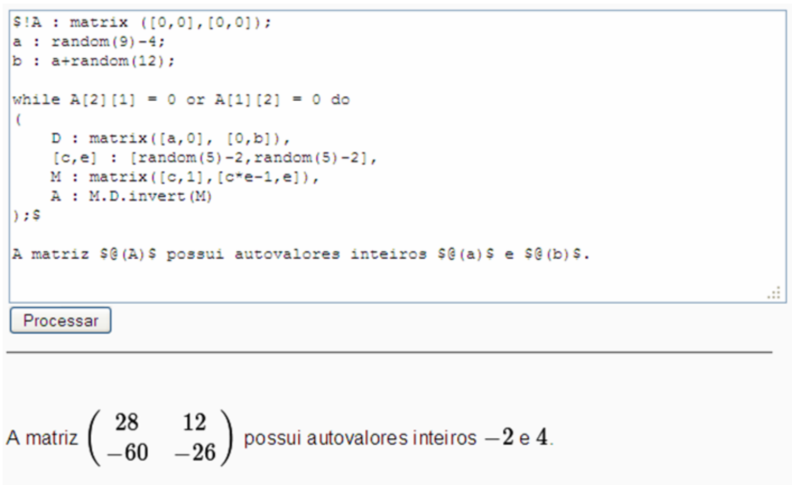


Figura 3.2: Geração de matrizes inteiras com autovalores inteiros

3.3.3 Escalonando Matrizes

Um tema que não costuma faltar nos cursos introdutórios de Álgebra Linear é o do reconhecimento da (in)dependência linear em um conjunto de vetores dados. Isto é realizado rearranjando os vetores em colunas de uma matriz, que sofre uma transformação para algum tipo de forma escalonada. Nesta forma, surgem zeros em posições da matriz que permitem concluir por inspeção visual se os vetores são ou não são linearmente dependentes.

Em um contexto correlato – o da resolução de sistemas lineares – muitos livros se limitam a apresentar um algoritmo mecânico de escalonamento para chegar às soluções do sistema ou constatar sua inexistência. Este algoritmo consiste na aplicação de uma sequência de operações sobre as linhas da matriz cujas entradas são os coeficientes das equações (LAY, 1997, [9]), incorporando a ela a coluna dos termos independentes: a matriz *aumentada*, como é normalmente chamada. Cada linha, durante o algoritmo, é substituída por uma combinação linear entre ela e alguma outra linha, de modo a ir produzindo, na matriz, as posições zeradas que se desejam.

Em um tratamento conceitual adequado, a ideia envolvida no processo de escalonamento de uma matriz A, conforme é descrita em (HOFFMAN & KUNZE, 1971, [7]), é a de obter uma sequência de matrizes $A = A_0, A_1, \ldots, A_p$, onde $p \leq n$. Cada matriz parcial A_i possui as i primeiras colunas satisfazendo: $A_{i_{ii}} = 1$ e $A_{i_{ki}} = 0$, para $k = i+1, \ldots, m$. Na etapa i do processo, existe uma matriz $\mathcal{E}_\rangle$, que multiplicada à esquerda de A_{i-1}, produz A_i. Ao final, teremos

$$A_p = \mathcal{E}_p \cdot \mathcal{E}_{p-1} \cdots \mathcal{E}_1 \cdot A,$$

onde A_p é a matriz A na forma escalonada.

Para exemplificar, consideremos a matriz

$$A = \begin{bmatrix} 3 & 2 & 13 \\ 1 & 0 & 3 \\ -2 & 1 & -4 \\ 2 & 1 & 8 \end{bmatrix}$$

e tratemos de escaloná-la passo a passo. O objetivo, na primeiro etapa, é concentrar-nos sobre a primeira coluna de A: gerar 1 na primeira posição e zerar o restante da primeira coluna, efetuando combinações lineares de linhas. Uma maneira (não única!) de obter este resultado é (1º) somar a terceira linha à quarta linha; (2º) somar a segunda linha à terceira dividida por 2; (3º) somar a primeira linha à segunda multiplicada por -3; e (4º) dividir a primeira linha por 3. A matriz que, multiplicada a esquerda de A, produz estes efeitos é precisamente:

$$\mathcal{E}_1 = \begin{bmatrix} \frac{1}{3} & 0 & 0 & 0 \\ 1 & -3 & 0 & 0 \\ 0 & 1 & \frac{1}{2} & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix}.$$

Logo,

$$A_1 = \mathcal{E}_1 \cdot A = \begin{bmatrix} 1 & \frac{2}{3} & \frac{13}{3} \\ 0 & 2 & 4 \\ 0 & \frac{1}{2} & 1 \\ 0 & 2 & 4 \end{bmatrix}.$$

O objetivo, a seguir, é trabalhar a segunda coluna de A_1, tornando a segunda posição igual a 1 e zerando a terceira e a quarta posições. Para isto, podemos: (1º) somar a terceira linha de A_1 à sua quarta linha multiplicada por $-\frac{1}{4}$; somar a segunda linha à terceira multiplicada por -4; e (3º) dividir a segunda linha por 2. A matriz que, multiplicada à esquerda de A_1, produz estes efeitos é:

$$\mathcal{E}_2 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \frac{1}{2} & 0 & 0 \\ 0 & 1 & -4 & 0 \\ 0 & 0 & 1 & -\frac{1}{4} \end{bmatrix}.$$

Logo:

$$A_2 = \mathcal{E}_2 \cdot \mathcal{E}_1 \cdot A = \begin{bmatrix} 1 & \frac{2}{3} & \frac{13}{3} \\ 0 & 1 & 2 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}.$$

Neste pequeno exemplo, o escalonamento terminou, revelando claramente que a terceira coluna não é linearmente independente das duas primeiras. Portanto a matriz

$$\mathcal{E} = \mathcal{E}_2 \cdot \mathcal{E}_1 = \begin{bmatrix} \frac{1}{3} & 0 & 0 & 0 \\ \frac{1}{2} & -\frac{3}{2} & 0 & 0 \\ 1 & -7 & -2 & 0 \\ 0 & 1 & \frac{1}{4} & -\frac{1}{4} \end{bmatrix},$$

multiplicando A pela esquerda, produzirá como resultado a matriz A_2, que é uma forma escalonada de A. É evidente que o processo assim descrito envolveu escolhas de combinações lineares específicas entre linhas a fim de obter os efeitos desejados nas diversas entradas da matriz; portanto, a matriz $\mathcal{E}$ obtida no exemplo não é a única possível.

Pelos questionamentos teóricos que esta abordagem permite, e que serão levantados um pouco mais adiante, é importante que o aluno perceba o escalonamento como resultado deste processo. Para isso, a obtenção da matriz $\mathcal{E}$ e das sequências de matrizes $\mathcal{E}_i$ e A_j deve ser explicitada durante a explanação do assunto.

Com o MathMoodle, a pedido do professor, elaboramos um script detalhando este processo. O desafio a vencer era que o Maxima disponibiliza a função $echelon(A)$, que retorna diretamente uma forma escalonada da matriz A; porém, não nos fornece nenhuma função que nos permita obter as matrizes parciais envolvidas.

Nossa atuação, sob orientação do professor, consistiu em, utilizando a função *echelon*, explicitar em um script estas matrizes intermediárias. Um raciocínio simples mostra que uma possível candidata a $\mathcal{E}$ que escalona A pode ser obtida concatenando-se à direita da matriz A a matriz identidade de ordem m:

$$\left[\begin{array}{ccc|cccc} 3 & 2 & 13 & 1 & 0 & 0 & 0 \\ 1 & 0 & 3 & 0 & 1 & 0 & 0 \\ -2 & 1 & -4 & 0 & 0 & 1 & 0 \\ 2 & 1 & 8 & 0 & 0 & 0 & 1 \end{array}\right].$$

Aplicando a esta matriz estendida a função *echelon*(A), provida pelo Maxima, obtemos

$$\left[\begin{array}{ccc|cccc} 1 & \frac{2}{3} & \frac{13}{3} & \frac{1}{3} & 0 & 0 & 0 \\ 0 & 1 & 2 & \frac{1}{2} & -\frac{3}{2} & 0 & 0 \\ 0 & 0 & 0 & 1 & -7 & -2 & 0 \\ 0 & 0 & 0 & 0 & 1 & \frac{1}{4} & -\frac{1}{4} \end{array}\right],$$

a submatriz formada pelas m últimas colunas é exatamente a matriz $\mathcal{E}$ procurada.

Utilizando este raciocínio, implementamos no Maxima o comando $elm(A)$ [2] e, na tabela de conversão de nomes do MathWriting, o comando correspondente $\backslash elm(A)$. Ele retorna a matriz $\mathcal{E}$, tal que $\mathcal{E} \cdot A$ seja igual a uma forma escalonada de A.

Com a disponibilidade do comando $\backslash elm$, foi possível construir o script requerido pelo professor. Ele está transcrito a seguir e é constituído por uma seção algorítmica e uma seção de texto. Note que é utilizada uma função que não existe no Maxima, chamada `pcols`, que está codificada diretamente na seção algorítmica do script. Esta função devolve a submatriz formada pelas n primeiras colunas de uma matriz M.

[2] Segundo colocação do professor, "elm" teria a ver com a palavra "elementar", que é normalmente o nome atribuído às matrizes que realizam as operações sobre as linhas.

```
$! A : matrix ([3,2,13],[1,0,3],[-2,1,-4],[2,1,8]);
pcols(M,n):=transpose(rest(transpose(M),n-length(transpose(M))));
C1: pcols(A,1);
A1: elm(C1).A;
C2: pcols(A1,2);
A2: elm(C2).A1;$

Dada a matriz $A = @(A)$, desejamos obter uma forma escalonada de $A$.
Seja $C_1 = @(C1)$ a submatriz formada pela primeira coluna de $A$.
$E_1$ = \elm($C_1$) = $\elm(C1)$ e $A_1 = E_1.A = @(A1)$
Seja C_2 = @(C2)$ a submatriz formada pelas duas primeiras colunas de $A_1$.
$E_2$ = \elm($C_2$) = $\elm(C2)$ e $A_2 = E_2.A_1 = @(A2)$.
$A_2$ é uma forma escalonada de $A$.
```

A visualização do processamento deste script encontra-se na Figura 3.3.

Dada a matriz $A = \begin{pmatrix} 3 & 2 & 13 \\ 1 & 0 & 3 \\ -2 & 1 & -4 \\ 2 & 1 & 8 \end{pmatrix}$, desejamos obter uma forma escalonada de A.

Seja $C_1 = \begin{pmatrix} 3 \\ 1 \\ -2 \\ 2 \end{pmatrix}$ a submatriz formada pela primeira coluna de A.

E_1 = \elm(C_1) = $\begin{pmatrix} \frac{1}{3} & 0 & 0 & 0 \\ 1 & -3 & 0 & 0 \\ 0 & 1 & \frac{1}{2} & 0 \\ 0 & 0 & 1 & 1 \end{pmatrix}$ e $A_1 = E_1.A = \begin{pmatrix} 1 & \frac{2}{3} & \frac{13}{3} \\ 0 & 2 & 4 \\ 0 & \frac{1}{2} & 1 \\ 0 & 2 & 4 \end{pmatrix}$

Seja $C_2 = \begin{pmatrix} 1 & \frac{2}{3} \\ 0 & 2 \\ 0 & \frac{1}{2} \\ 0 & 2 \end{pmatrix}$ a submatriz formada pelas duas primeiras colunas de A_1.

E_2 = \elm(C_2) = $\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \frac{1}{2} & 0 & 0 \\ 0 & 1 & -4 & 0 \\ 0 & 0 & 1 & -\frac{1}{4} \end{pmatrix}$ e $A_2 = E_2.A_1 = \begin{pmatrix} 1 & \frac{2}{3} & \frac{13}{3} \\ 0 & 1 & 2 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$.

A_2 é uma forma escalonada de A.

Figura 3.3: Visualização da saída produzida pelo script de escalonamento

Do ponto de vista didático, apresentação do tema com a correta ênfase conceitual permitiu ao professor investigar com a turma questões teóricas que, com a abordagem puramente mecânica e operacional, não teriam vez.

Pode-se demonstrar (HOFFMAN & KUNZE, 1971, [7]) que a matriz $\mathcal{E}$ que escalona A é invertível (porque ela é o produto de matrizes elementares, todas invertíveis). Portanto, a forma escalonada de A consiste no produto de uma matriz invertível por A.

Sendo M uma matriz $m \times n$, designemos por $T_M : \mathbb{R}^n \to \mathbb{R}^m$ a transformação linear dada por $x \mapsto Mx$. Sob esta ótica, a forma escalonada de A corresponde à transformação linear composta $T_{\mathcal{E}} \circ T_A$. Sendo $\mathcal{E}$ invertível, $T_{\mathcal{E}}$ é injetiva e a composta $T_{\mathcal{E}} \circ T_A$ preserva algumas características de T_A (o posto, por exemplo). Cabem, portanto, as seguintes reflexões junto aos alunos:

- Quais subespaços (núcleo? imagem?) de T_A são preservados na composição $T_{\mathcal{E}} \circ T_A$?
- Por que a relação de dependência linear entre as colunas de A permanece a mesma em $\mathcal{E} \cdot A$?
- A matriz $\mathcal{E}$ é construída através do produto de matrizes elementares que efetuam operações sobre as linhas de A. No entanto, multiplicar $\mathcal{E}$ à esquerda de A tem o efeito de agir sobre as colunas de A. Sendo $\mathcal{E}$ invertível, ela pode ser interpretada como uma matriz de mudança de base em $\mathbb{R}^m$, que é o contradomínio de T_A. Que resultados podem ser explorados a partir desta constatação?

Pode-se questionar, e com toda razão, que não é necessário dispor do MathMoodle para fazer tais reflexões. Em cursos mais avançados de Álgebra Linear, onde o enfoque é teórico apenas, estas questões são respondidas sem apelo a qualquer cálculo. Isto é fato, mas, em um curso destinado à graduação, explicitar as contas faz parte do programa e a facilidade de calcular com o MathMoodle torna-se uma motivação para levantar questões conceituais, associadas a uma gama de exemplos que podem ser construídos de forma imediata.

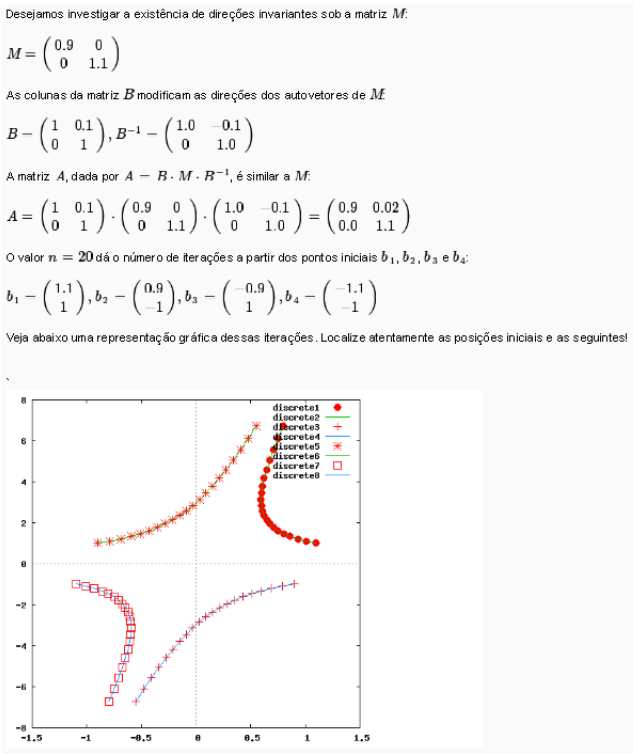
Outra observação importante que surgiu ao longo deste desenvolvimento é que, com o MathMoodle, é possível escrever comandos específicos para um curso, como foi feito com o comando $\backslash elm$ neste curso. Por enquanto, esta possibilidade requer a intervenção do administrador, principalmente no caso citado, em que foi necessário acrescentar a função também ao Maxima.

3.3.4 Identificando Graficamente Direções Invariantes sob um Operador Linear

Outro tema importante que recebeu tratamento especial neste curso foi o tópico de autovalores e autovetores.

Dado um operador linear T em $\mathbb{R}^2$, uma forma interessante, sugerida pelo professor, para ilustrar geometricamente a existência (ou não!) de subespaços invariantes unidimensionais sob T, seria escolher uma base β em R^2 e um ponto inicial P e, iteradamente, aplicar a P a matriz $A = [T]_\beta$, [3] obtendo uma sequência de pontos $P, A \cdot P, A^2 \cdot P, \cdots, A^n \cdot P$, $n > 0$.

Direções invariantes podem destacar-se nitidamente quando esta sequência de pontos é mostrada em um gráfico, o que pode servir como motivação para começar a abordar o tópico a partir de questionamentos sobre o conceito chave, que é a invariância. Na Figura 3.4, temos a ilustração.


Desejamos investigar a existência de direções invariantes sob a matriz M:

$$M = \begin{pmatrix} 0.9 & 0 \\ 0 & 1.1 \end{pmatrix}$$

As colunas da matriz B modificam as direções dos autovetores de M:

$$B = \begin{pmatrix} 1 & 0.1 \\ 0 & 1 \end{pmatrix}, B^{-1} = \begin{pmatrix} 1.0 & -0.1 \\ 0 & 1.0 \end{pmatrix}$$

A matriz A, dada por $A = B \cdot M \cdot B^{-1}$, é similar a M:

$$A = \begin{pmatrix} 1 & 0.1 \\ 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} 0.9 & 0 \\ 0 & 1.1 \end{pmatrix} \cdot \begin{pmatrix} 1.0 & -0.1 \\ 0 & 1.0 \end{pmatrix} = \begin{pmatrix} 0.9 & 0.02 \\ 0.0 & 1.1 \end{pmatrix}$$

O valor $n = 20$ dá o número de iterações a partir dos pontos iniciais b_1, b_2, b_3 e b_4:

$$b_1 = \begin{pmatrix} 1.1 \\ 1 \end{pmatrix}, b_2 = \begin{pmatrix} 0.9 \\ -1 \end{pmatrix}, b_3 = \begin{pmatrix} -0.9 \\ 1 \end{pmatrix}, b_4 = \begin{pmatrix} -1.1 \\ -1 \end{pmatrix}$$

Veja abaixo uma representação gráfica dessas iterações. Localize atentamente as posições iniciais e as seguintes!

Figura 3.4: Direções invariantes sob um operador linear

Este desenvolvimento nos remete a questionamentos que, realizados em aula, ajudam os alunos a entender as propriedades envolvidas neste tema. Assim o professor pode apresentar as seguintes indagações aos estudantes:

- O que acontece se o operador tem dois autovalores reais distintos?
- O que acontece se estes autovalores reais coincidem?
- O que acontece se eles são complexos?

O efeito das alterações nos dados, quer modificando as entradas da matriz quer

[3] $[T]_\beta$ significa, aqui, a matriz do operador T na base β

mudando o ponto de partida produz resultados facilmente visíveis e que contribuem para um maior entendimento conceitual do tema.

O sistema de coordenadas pode também ser alterado, por meio de uma transformação de similaridade. Examinar os mesmos exemplos em diferentes sistemas de coordenadas, bem como aprender a escolher um sistema adequado para melhor expressar a solução de um problema, são pontos fundamentais da Álgebra Linear que nem sempre podem ser trabalhados nos cursos. Todos os cálculos que, ao quadro-negro, tomariam muito tempo de aula, puderam, com o MathMoodle, ser realizados de forma imediata com alguns comandos simples.

3.4 Conclusões sobre o Experimento de Validação

Com o intuito de validar, do ponto de vista didático, as ferramentas que havíamos desenvolvido, o curso de Álgebra Linear II mostrou-se um campo fértil que aponta muito claramente para a conclusão de que o MathMoodle e o mwEdPlayer são instrumentos computacionais cujo uso criterioso e planejado pode beneficiar e enriquecer o ensino desta disciplina.

Relatamos, nas seções anteriores deste capítulo, apenas os pontos principais, em que a intervenção das ferramentas no curso se deu de modo mais ostensivo. Porém, mesmo em tópicos que não haviam sido selecionados previamente para este estudo, o MathMoodle e o mwEdPlayer acabaram sendo utilizados em aulas e fora delas, devido às facilidades que introduziam.

Os scripts revelaram-se um poderoso instrumento do ponto de vista didático, reforçando para a nossa pesquisa que as ferramentas desenvolvidas efetivamente tinham o efeito pretendido de complementar o ensino da disciplina de maneira proveitosa tanto para o professor quanto para os monitores e alunos. A possibilidade de disponibilizá-los para posterior trabalho extraclasse através do Módulo Scripts constituiu um importante acessório, na medida em que incentivou os alunos a modificar por conta própria os cenários apresentados em aula, obtendo suas conclusões num processo de estudo dirigido elaborado pelo professor, com a nossa ajuda.

Os tópicos escolhidos (escalonamento e autovalores), que recebem normalmente nos cursos um tratamento com privilégio do operacional, puderam ser trabalhados sem perder de vista os conceitos algébricos que os fundamentam, dando ao professor o espaço necessário para levantar, do ponto de vista teórico, as questões relevantes que apontamos e que normalmente são omitidas nas abordagens tradicionais desta disciplina em graduação.

A possibilidade de modificar dados em um script e reprocessá-lo em seguida teve um efeito muito positivo nas aulas que acompanhamos. A riqueza de cenários que se consegue fornecer acerca de um tema motiva os alunos a questionamentos, colocações e dúvidas durante a aula que nem sempre costumam aparecer quando os assuntos são tratados no quadro-negro com recurso a poucos exemplos. No tópico de autovalores, isto foi bastante visível, pois pequenas modificações nas entradas de uma matriz (mesmo no caso 2×2) mudam completamente o comportamento do espectro, o que provoca a surpresa e desperta a curiosidade dos alunos. Numa aula tradicional, raramente se consegue este efeito.

Durante o experimento, como resultado do acompanhamento sistemático das aulas e com a continuidade do uso das ferramentas, diversos aprimoramentos e correções tiveram de ser feitos. A própria ferramenta mwEdPlayer não existia quando o curso foi iniciado: ela foi fruto da observação direta em sala de aula de que não sobrava espaço suficiente na tela para exibir simultaneamente o texto de um script e o resultado de seu processamento. O Módulo Scripts foi também um desenvolvimento que ocorreu ao longo do curso, quando, a pedido dos próprios alunos, o professor decidiu disponibilizar os scripts que eram mostrados em sala.

Ficou claro para nós que várias melhorias ainda são necessárias, considerando que as ferramentas devem ser úteis também a outros cursos de matemática e não somente ao curso de Álgebra Linear. A introdução do comando $\backslash elm$, por exemplo, foi trabalhosa: ela exigiu um tipo de intervenção direta na plataforma de um modo que ainda não é possível ao instrutor de um curso. Foi preciso acrescentar esta função ao arquivo de inicialização do Maxima, de modo que, à semelhança das funções nativas, ela estivesse sempre disponível e não precisasse ser reescrita em cada script. Para um uso futuro, avaliamos que esta possibilidade de introduzir funções permanentes no CAS deva ser oferecida ao professor.

Um outro ponto não menos importante a ressaltar neste experimento foi o envolvimento dos monitores no processo. A motivação e o entusiasmo demonstrados por esta equipe de alunos recém egressos da disciplina contagiou positivamente os colegas que a estavam cursando. Um dos problemas apontados por (SILVA, 2009, [12]) na observação da versão preliminar da plataforma no curso de Cálculo II foi justamente o do pouco interesse demonstrado por parte dos estudantes em aprender e utilizar a ferramenta, em parte devido aos erros e dificuldades de uso que ela apresentava, dado seu estágio de desenvolvimento ainda muito incipiente.

Com as melhorias introduzidas e a cooperação dos monitores, pudemos observar uma realidade oposta neste nosso experimento: constatamos intensa atividade colaborativa entre monitores e alunos, na elaboração das entradas de glossário, no fórum, no chat, comentando soluções de exercícios, adaptando e postando na plataforma enunciados colhidos da bibliografia indicada, formulando scripts de solução e até mesmo propondo novos exercícios.

Para os monitores, inclusive, ocorreu um verdadeiro processo de aprimoramento dos seus conhecimentos. Muitas vezes, ao explicarem a solução de um exercício ou ao completarem a definição numa entrada de glossário, eles cometiam falhas conceituais que eram discutidas nas reuniões com o professor, por nós observadas.

Capítulo 4

Conclusões

A partir de observações relatadas na dissertação de mestrado (SILVA, 2009, [12]) e do acompanhamento paralelo que nós próprios fizemos do curso de Cálculo II, constatamos que era necessário submeter o MathMoodle a uma considerável revisão, caso realmente desejássemos torná-lo uma ferramenta profissional que pudesse ser utilizada em cursos onde a expressão e a comunicação de conteúdo matemático estivessem presentes, obtendo algum ganho de aprendizagem com este uso.

De início, era preciso eliminar as deficiências que foram observadas durante o uso da versão preliminar, que acabavam por desviar o foco, por parte dos alunos, do conteúdo ensinado para o entendimento dos pormenores de utilização da ferramenta, o que acarretou a descontinuidade de seu uso ao longo do referido curso. Poucos usuários, em geral, dispõem-se à leitura de um manual e esperar de alunos esta motivação é algo totalmente irreal. Precisávamos, portanto, conceber um novo processo de validação em que a assimilação dos detalhes de uso da ferramenta fosse paulatina e indolor, de maneira a não desmotivar os alunos e não desviar seu foco do aprendizado do conteúdo, que é o verdadeiro objetivo de um curso.

Ao final de 2010, portanto, possuíamos uma ferramenta totalmente revista e reimplementada, que necessitava de dois tipos de validação. O primeiro, obviamente, é a espécie de teste que se espera de qualquer sistema computacional e que consiste em submetê-lo ao uso sucessivo de grupos cada vez menos supervisionados, até poder homologá-lo em definitivo. Um segundo tipo de validação diz respeito ao propósito essencial da ferramenta: servir de coadjuvante no ensino de disciplinas regulares de cunho matemático, em princípio, na graduação em carreiras técnicas.

Tínhamos, então, estas duas metas a cumprir. O professor da disciplina dispôs-se gentilmente a ministrar, por dois períodos letivos sucessivos do ano de 2011, a disciplina de Álgebra Linear II, que serviria de palco a nosso experimento de validação. Nossa participação neste processo de validação consistiria em observar as aulas ministradas, em especial aquelas em que a interferência do MathMoodle havia sido prevista, acompanhar as atividades dos monitores e alunos no MathMoodle e, é claro, corrigir os erros de implementação que se revelavam ao longo do uso e agregar novas funcionalidades, conforme se mostrassem necessárias.

As melhorias sintáticas introduzidas, juntamente com a dinâmica e as necessidades percebidas em sala de aula, motivaram o surgimento dos novos modos de interação com o Maxima: o modo silencioso e o modo literal. Em decorrência destes acréscimos, a organização de textos matemáticos como scripts e os benefícios percebidos com a adoção deste estilo de escrita levaram-nos à criação, no Moodle, do Módulo Scripts.

Os scripts foram, sem dúvida, a melhor contribuição resultante deste processo de validação para a exposição do conteúdo. A possibilidade de o instrutor disponibilizar aos participantes de um curso verdadeiros roteiros de estudo, intercalando seções algorítmicas e textuais que podem ser alteradas em direção a novos cenários, cumpriu um duplo papel na experiência: ao mesmo tempo que os scripts funcionavam como manual de uso, do ponto de vista sintático, para os alunos e monitores, através do já descrito processo de colagem e adaptação, eles também constituíram um valioso complemento ao estudo, suprindo as lacunas das fontes bibliográficas recomendadas, provendo exercícios e incentivando a aprendizagem autônoma.

Ao fim do experimento, podemos afirmar que temos a ferramenta validada nos dois aspectos que mencionamos, pelos motivos que expomos a seguir.

De um lado, a revisão empreendida e as correções dos erros apontados durante o uso permitiram-nos atingir um sólido patamar de depuração, em que as instabilidades praticamente desapareceram. É claro que a validação computacional realizada deu-se em um grupo ainda muito supervisionado, no qual muitos erros podem ter sido evitados exatamente pelo direcionamento que induzimos ao uso. É necessária ainda uma etapa adicional de testes, que permitirão homologar em definitivo a nossa contribuição, disponibilizando-a a outros grupos interessados.

Por outro lado, acompanhamos de perto os benefícios introduzidos no ensino da disciplina de Álgebra Linear II com o uso do MathMoodle, que já relatamos na seção final do Capítulo 3. A dificuldade apontada por alguns professores em graduar os aspectos teóricos e práticos da disciplina, em função de especificidades do público alvo e da reduzida carga horária (4 horas semanais apenas), foi certamente atenuada no contexto que observamos, conforme expomos: tópicos que são abordados normalmente de um ponto de vista puramente operacional puderam, com as facilidades de cálculo providas pelo MathMoodle, ter seu respaldo conceitual também explorado durante o curso.

O exame analítico da ferramenta que obtivemos revelou que a essência de sua estrutura, constituída por três agentes, possuía certa abrangência, no sentido de poder concretizar-se em realidades computacionais diversas. Este fato nos motivou a denominar o que tínhamos em mãos não apenas como uma mera integração de programas já existentes, mas como uma tecnologia para redação matemática que, no cenário observado, instanciava-se como a ferramenta MathMoodle. Além desta possível materialização computacional, mencionamos também a ferramenta mwEdPlayer e o sistema AtenaME, instâncias desta mesma tecnologia, ambos desenvolvidos no LIMC/UFRJ.

Na direção de trabalhos futuros, podemos destacar algumas frentes:

- A incorporação de novas funcionalidades à ferramenta. Uma delas, apontada no Capítulo 3, seria habilitar o instrutor de um curso do Moodle a instalar funções permanentes no Maxima, de maneira que elas não precisem ser replicadas a cada script em que são invocadas. Fizemos isto manualmente com a função $\backslash elm$, mas julgamos que esta capacidade deva ser repassada ao instrutor.

  Desta maneira, ele poderá estabelecer um repertório específico e delimitado de recursos para um determinado curso. Isto possibilita a formulação de exercícios especiais, em que, mais do que a solução de um certo problema, o professor está interessado em que o aluno o resolva utilizando estritamente os recursos especificados, que funcionam como uma espécie de "caixa de utensílios". Dependendo da finalidade e do escopo do curso, desenvolver no aluno este tipo de habilidade (resolver problemas com os meios de que se dispõe) pode ser primordial na formação que se pretende.

- A interação entre o Maxima e o programa Gnuplot, um gerador de gráficos, foi muito pouco explorada nos scripts do curso. Apenas em um contexto específico, que investigava as direções invariantes sob um operador linear, houve a geração de gráficos.

  Da maneira como está disponibilizada hoje, esta facilidade ainda está muito distante de poder ser aproveitada pelo usuário não versado na sintaxe do Maxima. Basta observar o exemplo mostrado na Figura 3.4 para constatar o que estamos afirmando. Pensamos que um certo investimento deva ser realizado em criar alternativas sintáticas mais simples e acessíveis para a geração de gráficos, que são um recurso tão importante no ensino da matemática e ciências.

- O desenvolvimento de um conjunto adicional de scripts envolvendo algoritmos e textos, que sejam interessantes do ponto de vista matemático, mas que também cumpram a função de um manual de uso, em que a assimilação da sintaxe proposta e dos recursos do CAS, por parte de professores e estudantes, se dê de forma mais atraente. Com o sucesso dos scripts que observamos durante o curso de Álgebra Linear, conjecturamos que esta seria uma boa alternativa ao tradicional "manual do usuário".

- A divulgação mais ampla da ferramenta dentro do próprio meio universitário, junto a professores que ministram disciplinas que exijam, de alguma forma, a redação de conteúdo matemático com qualidade.

- Um projeto de uso e divulgação do MathMoodle nos ensinos fundamental e médio. É claro que um empreendimento deste vulto necessitaria da participação de especialistas em ensino de matemática e não seria um trabalho a ser empreendido por nós apenas.

Referências Bibliográficas

[1] AXLER, S., *Linear Algebra Done Right*, Springer, 1997.

[2] DEVOLDER, R. G., *MathWriting: Escrita e Comunicação Matemática em Ambientes Virtuais para o Ensino à Distância.* Monografia, Rio de Janeiro: UFRJ/IM, 2008.

[3] DOUGIAMAS, M., TAYLOR, P., Moodle: Using Learning Communities to Create an Open Source Course Management System. In P. Kommers & G. Richards (Eds.), *Proceedings of World Conference on Educational Multimedia, Hypermedia and Telecommunications 2003*, (pp. 171-178). Chesapeake, VA: AACE., 2003.

[4] GONÇALVES, A.; SOUZA, R. M. L. *Introdução à álgebra linear.* São Paulo: Editora Edgar Blucher Ltda, 1977.

[5] GUIMARÃES, L. C., VERNET, O., DEVOLDER, R. G., Integrating Computer Algebra and Course Management Systems for Undergraduate Mathematics, *12th International Congress on Mathematical Education*, Seoul, 2012, Disponível em `http://www.icme12.org/upload/UpFile2/TSG/1421.pdf`, Acessado em 16 de julho de 2012.

[6] HALMOS, P.R., *Finite-Dimensional Vector Spaces*, Undergraduated Texts in Mathematics, Springer, 1974.

[7] HOFFMAN, K., KUNZE, R., *Linear Algebra*, Prentice-Hall, 1971.

[8] JIPSEN, P., *ASCIIMathML: Math on the web for everyone*, Chapman University, 2007. Disponível em `http://www1.chapman.edu/~jipsen/mathml/asciimath.html`, Acessado em 20 de junho de 2012.

[9] LAY, D. *Álgebra Linear e suas Aplicações*, 2ª edição, Editora LTC. 1997.

[10] MAXIMA, Maxima 5.27.0 Manual, 2012, Disponível em `http://maxima.sourceforge.net/docs/manual/en/maxima.html`, Acessado em 20 de junho de 2012.

[11] MOORE, M., *Moodle Open Source Course Management System: A Free Alternative to Blackboard*, IT899 Masters Project in Instructional Design and Technology, Dr. Armand Seguin, Dr. Harvey Foyle, Dr. Jane Eberle, 2003.

[12] SILVA, Ulisses D. *MathMoodle - Estudo de Caso sobre um CMS Munido de Ferramentas de Computação Simbólica e de Comunicação de Conteúdo Matemático*, Dissertação de mestrado, Rio de Janeiro: UFRJ/IM, 2009.

[13] SOUZA, P. N., FATEMAN, R. J., MOSES, J., YAPP, C. *The Maxima Book*, 2004. Disponível em `http://maxima.sourceforge.net/docs/maximabook/maximabook-19-Sept-2004.pdf`, Acessado em 20 de junho de 2012.

[14] STEINBRUCH, A., WINTERLE, P. *Introdução à Álgebra linear*, São Paulo: Makron Books, 1990.

[15] VERNET, O., AtenaME – Manual do Usuário, Disponível em http://profpapier.limc.ufrj.br/AtenaME, Acessado em 16 de julho de 2012.

[16] WAWRO, M., SWEENEY, G., RABIN, J. M., Subspace in Linear Algebra: Investigating Students Concept Images and Interactions with the Formal Definition. *Educational Studies in Mathematics*, 78, 2011, pp. 1-19.

Apêndice A

Comandos do MathWriting

@: obtém o valor da operação ou constante dada.
abs: obtém o valor absoluto do número dado.
adcoluna: adiciona uma ou mais colunas a uma matriz predefinida.
adjunta: retorna a matriz adjunta de uma matriz.
adlinha: adiciona uma ou mais linhas a uma matriz predefinida.
antiderivada: calcula a antiderivada de uma expressão em relação a variável escolhida.
autovalores: obtém os autovalores, em forma de lista, de uma matriz.
autovetores: obtém os autovetores de uma matriz.
autovetorunitario: obtém os auto vetores unitários de uma matriz.
coluna: obtém uma coluna de uma matriz.
comparar: obtém um operador de comparação relativo aos números dados.
cos: obtém o coseno do número dado.
denom: retorna o denominador da expressão, que é retornada caso não seja uma razão.
derivar: calcula a derivada de uma expressão. é possível selecionar ordem e variaveis.
determinante: obtém o determinante de uma matriz.
distribuir: distribui adições sobre produtos.
dividir: retorna o quociente da divisão dos dois polinômios passados.
escalonar: constrói uma nova matriz escalonando a matriz predefinida.
escalonar: escalona uma matriz.
exp: obtém a exponencial do número dado.
expandir: expande a expressão dada.
fatorar: fatora a expressão dada.
fracao: coloca todos os termos já calculados sobre um denominador comum.
fracoesparciais: expande a expressão na forma de frações parciais em relação a

variável escolhida.
funcaomatriz: executa uma função predefinida em uma matriz predefinida.
geramatriz: inverte a matriz predefinida.
gfatorar: fatora a expressão sobre os inteiros de gauss, ou seja, com a unidade i imaginária.
gradiente: obtém o vetor gradiente de uma função.
ident: gera uma matriz identidade n por n.
impar: obtém verdadeiro se o parâmetro for um inteiro ímpar e falso para os demais casos.
integrar: calcula a integral de uma expressão. é possível definir o intervalo de integração.
inversa: inverte a matriz predefinida.
limite: calcula o limite da expressão quando a variável escolhida tende ao valor escolhido.
linha: retorna uma linha de uma matriz.
log: obtém o logaritmo do número dado.
maiorexp: !maiorexp(p1, p2, y) procura entre p1 e p2 o maior expoente da letra y.
matriz: define uma matriz através dos elementos passados nas listas.
matrizcoef: gera a matriz dos coeficientes de um sistema de equações lineares.
matrizcoefaum: cria a matriz aumentada dos coeficientes de um sistema de equações lineares.
matrizdiag: cria uma matriz diagonal n por n, com elementos da diagonal iguais a x.
matrizzero: cria uma matriz n por m com todos os elementos iguais a 0.
max: obtém o maior valor entre os dados, é possível acrescentar mais valores.
menor: obtém a submatriz excluindo a linha i e a coluna j.
menorexp: !menorexp(p1, p2, y) procura entre p1 e p2 o menor expoente da letra y.
min: obtém o menor valor entre os dados, é possível acrescentar mais valores.
mod: obtém o resto da divisão do primeiro valor pelo segundo valor.
num: retorna o numerador da expressão, que é retornada caso não seja uma razão.
ordenar: ordena uma lista de números conforme o operador no segundo parâmetro.
permanente: calcula o permanente de uma matriz.
piso: obtém o número inteiro mais próximo menor ou igual ao número dado.
pisoraiz: calcula, por falta, a raiz quadrada do número dado.
policarac: retorna o polinômio característico de uma matriz.
posto: calcula o posto de uma matriz.
prodescalar: obtém o produto escalar das matrizes predefinidas.
produtorio: obtém o valor do produtório da expressão, em n, de a até b.

raiz: calcula a raiz quadrada do número dado.
resolver: resolve a(s) equação(ões) dada. recebe até 2 listas para decidir em que variáveis operar.
resolvesistema: resolve a lista de equações na lista de variáveis dada.
resto: obtém o resto da divisão dos dois polinômios passados.
sen: obtém o seno do número dado.
simplificar: simplifica a expressão dada, importante passar uma única razão.
somatorio: retorna o valor do somatório da expressão, em n, de a até b.
substituir: !substituir(a,b,c) substitui a expressão 'b', por 'a' em 'c' .
tan: obtém a tangente do número dado.
termo: obtém o iésimo termo de uma lista.
teto: obtém o número inteiro mais próximo maior ou igual ao número dado.
traco: obtém a soma dos elementos da diagonal principal de uma matriz.
transposta: gera a matriz transposta de uma matriz.
triangularizar: gera a maior forma triangular de uma matriz.
valor: obtém o valor da operação ou constante dada.
vetorcoluna: gera o vetor coluna com os elementos da lista passada como parâmetro.