ARQB30 – Modelagem da Informação da Cidade.

Python e Ferramentas para Georreferenciamento, Aquisição e Manipulação de Dados Geoespaciais.

Fernando Ferraz Ribeiro[1]

Resumo: O resumo deve apresentar

Palavras-chave: GIS. BIM. CIM. Ciência de Dados. Python.

1 INTRODUÇÃO

A aquisição, manipulação, armazenamento e criação de valor a partir de dados vem sendo reconhecida como uma das grandes contribuições da computação atual para o desenvolvimento do conhecimento humano. No campo dos dados geoespaciais não tem sido diferente. A academia e a prática vem produzindo e discutindo aplicações da Ciência de Dados na prestação de serviços para os moradores e visitantes (LIM; KIM; MAGLIO, 2018), no monitoramento do trânsito (BERMUDEZ-EDO; BARNAGHI; MOESSNER, 2018) e no planejamento das cidades (OSMAN, 2019).

Um outro importante paradigma que utiliza a criação de um modelo de informação, orientado ao estudo do espaço construído é o de *City Information Modeling* (CIM) (AMORIM, 2015; GIL; ALMEIDA; DUARTE, 2011; TAH; OTI; ABANDA, 2017; XU et al., 2014), que, do ponto de vista dos dados, pode ser definido como a construção, armazenamento e manutenção de bancos de dados com informações geoespaciais, semânticas e geométricas das cidades.

Enquanto a Ciência de Dados aborda a obtenção de dados, a transformação de dados em informação e a geração de valor, independente do formato e volume dos dados (EMC EDUCATION SERVICE, 2015), O CIM advoga pela criação de um

[1] *Fernando Ferraz Ribeiro. Arquiteto e Urbanista, Mestre em Modelagem Computacional.Professor na Faculdade de Arquitetura – UFBA. Pesquisador no LCAD.*

banco de dados específico, que armazenam um “modelo digital de cidades”, aplicado tanto para a análise de dados, quanto para o registro de modelos de informação e geometria das edificações e demais elementos da infraestrutura urbana. Os modelos CIM sugerem uma analogia com os modelos e conceitos de *Building Information Modeling* (BIM). Neste contexto, a interoperabilidade entre os formatos IFC, principal formato ligado ao BIM e CityGML, um esquema XML para representação de geometria e informações de elementos e infraestrutura das cidades, ganha destaque nas pesquisas da área.

De acordo com a revisão de literatura realizada por Ma e Ren (2017), analisando os estudos realizados entre 2008 e 2016, o pico desta pesquisa é alcançado em 2015, decaindo no ano seguinte. Este decrécimo, no entanto, é contrabalanceado pelo empenho em integrar a leitura de IFC na versão 3.0 do CityGML, em desenvolvimento desde 2014.. Embora o software ArchGis seja a ferramenta mais utilizada neste processo (31,7 %), o segundo grupo de ferramentas mais utilizado nesta integração são as ferramentas desenvolvidas pelos pesquisadores através de linguagens de programação (19,5%).

A programação e a análise e dados são habilidades consideradas fundamentais em várias áreas do conhecimento e atividades profissionais. A valorização destas habilidades tende a crescer no ambiente de pesquisa e na prática profissional nos próximos anos. Um relatório encomendado pela divisão de pesquisa e treinamento da empresa de software Oracle (BURNING GLASS TECHNOLOGIES, 2016) aponta que: Além do mercado de tecnologia da informação, trabalhar com programação tem sido uma característica profissional buscada em empregos ligados à engenharia, design e artes, análise de dados e postos de pesquisa científica; os empregos relacionados com programação crescem mais que o próprio mercado de trabalho e pagam melhores salários.

O presente trabalho foca em algumas bibliotecas da linguagem Python relacionadas com dados geoespaciais. A leitura e escrita de arquivos, a integração com bancos de dados geoespaciais, a gravação e leitura de arquivos IFC e CityGML e as possibilidades futuras e aplicações atuais destas ferramentas. Apresentando

testes preliminares de algumas bibliotecas. Por fim os resultados e possibilidades são discutidas.

2 AMBIENTE PYTHON E DADOS GEOESPACIAIS

O Python é uma linguagem de programação de propósito geral, criada no início da década de 1990 por Guido van Rossum, mantida e desenvolvida pela fundação *Python Software Foundation*. Possui sintaxe simples e de fácil aprendizado, sendo uma das linguagens mais utilizadas no ensino e aprendizado de programação. Também é vastamente aplicada em pesquisa de ponta em computação, tendo a análise de dados e aprendizado de máquina como campos importantes para o impulsionamento e disseminação do seu uso.

Diversas aplicações CAD, GIS e BIM são desenvolvidas e/ou expandidas pela linguagem Python (Autocad, Rhinocesros, Grasshopper, Qgis, ArcGis, Revit, Dynamo, FreeCad, Belnder) tronando-a bastante utilizadas entre arquitetos e engenheiros, dentre outros profissionais. Além da facilidade do aprendizado, o Python é considerada uma linguagem de rápida implementação, extremamente concisa e consistente (RAMALHO, 2015).

Esta pesquisa realizou testes preliminares com o ambiente de programação Python, utilizando a distribuição científica Anaconda e o ambiente de desenvolvimento Jupyter. Apresentados na primeira seção. Em seguida a biblioteca Geopandas é apresentada, os experimentos realizados com a biblioteca são descritos e sua integração com as bibliotecas de aprendizado de máquina são apresentados. Na terceira seção a biblioteca Osmnx, responsável pela leitura de informações da base de dados OpensStreetMap é apresentada. Na quarta seção, apresentamos as bibliotecas utilizadas para leitura e escrita de bancos de dados, focando nos bancos de dados geoespaciais PostGis.

2.1 Ambiente Conda e Jupyter Notebooks.

Uma linguagem de programação é primeiramente definida em um documento de requisitos. Este documento detalha como a linguagem será implementada em detalhes. Quais os tipos de dados e variáveis, quais as palavras reservadas e funções embutidas, como funciona a sintaxe da linguagem, etc. A linguagem Python não é diferente. Seus requisitos e definições são mantidos pela fundação *Python Software Foundation,* e cada implementação da linguagem é chamada de distribuição. A distribuição padrão da linguagem é chamada de CPython e é disponibilizada pela mesma fundação. Outra importante distribuição é a chamada IronPython, desenvolvida pela Microsoft, ela permite a criação de programas em Python (versão 2.7x) para o f*ramework*.Net. Esta implementação é utilizada pelas interfaces de programação de diversos programas CAD e BIM (Autocad, Revit, Rhinoceros, Dynamo, etc).

No campo da programação científica a distribuição Anaconda (www.anaconda.com) apresenta elevado destaque. A distribuição utiliza o interpretador CPython adicionado de uma gama de bibliotecas científicas amplamente utilizadas, uma ferramenta de gerenciamento de ambiantes de programação, instalação de pacotes, ferramenta de gerenciamento gráfico e interfaces de programação. Disponível nas plataformas Windows, Linux e MacOs e possui mais de onze milhões de usuários em todo o mundo.

A maior vantagem de se utilizar a distribuição Anaconda é o sistema de gerenciamento de ambiantes. Referidos na documentação com *environments*, os ambientes de programação gerenciam as bibliotecas e suas dependências. A maioria das linguagens de programação não disponibilizam todas as suas funções diretamente no compilador ou interpretador. Muitas das funções são acessadas por pacotes (bibliotecas) desenvolvidos por terceiros.

É comum que os desenvolvedores destes pacotes utilizem outros pacotes para implementar novas funções. Se um pacote utiliza uma ou mais bibliotecas, oriundas de um ou mais pacotes para implementar suas funções diz-se que os pacotes utilizados na implementação são dependências do pacote que as utiliza. Os pacotes são desenvolvidos por versões e, nem sempre, as bibliotecas que

dependem de outras, não são necessariamente compatíveis com todas as versões de suas dependências. O sistema de gerenciamento de pacotes da distribuição Anaconda consegue criar ambientes de programação compatíveis e replicáveis que ajudam os programadores a trabalhar em instalações idênticas e controladas, resolvendo os problemas de conflito entre os diferentes pacotes utilizados pelo ambiente. Pelos testes realizados, o gerenciamento de pacotes por linha de comando mostrou-se mais eficiente e melhor documentado do que o ambiente gráfico de gerenciamento.

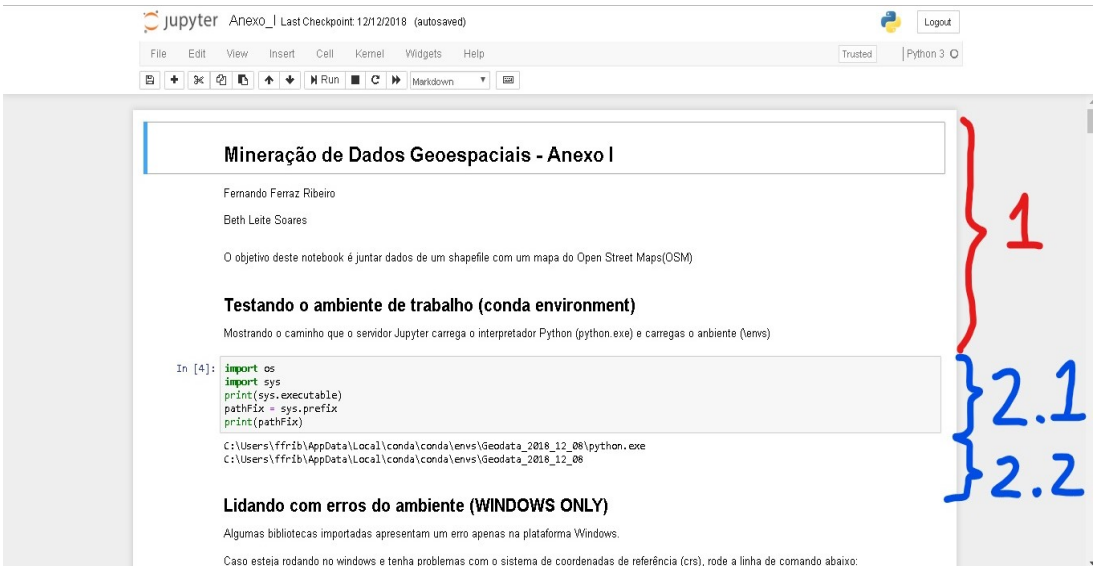

Figura 1: IDE Jupyter

Selecionando os pacotes de um determinado ambiente, é aconselhável utilizar uma IDE. Acrônimo de Integrated Development Environment, as IDE são utilizadas para facilitar o trabalho de programação, testes, acesso à ajuda e documentação, controle de versões dos arquivos de trabalho e uma série de tarefas correlatas ao trabalho de desenvolvimento de programas e *scripts*. A IDE escolhida para os testes foi a Jupyter, largamente utilizada em ciência de dados. Tratada como uma *analical sandbox* ou um *workspace* pela literatura da área (EMC EDUCATION SERVICE, 2015) pela facilidade de manipulação dos dados que este tipo de interface proporciona.

Os arquivos desta IDE são chamados de *notebooks* já que simulam um caderno de cálculos. Esses arquivos são editados através de um navegador de internet, podendo acessar ambientes e dados em servidores remotos ou locais. Existem dois tipos de células (campos onde pode se digitar textos) nos *notebooks:* Markdown e código. Na Figura 1 vemos algumas células marcadas com o número 1, estas correspondem as células do tipo Markdown. Estas células são utilizadas para inserir e formatar infirmações sobre o código na linguagem de formatação Markdown. As do tipo código são usadas para digitar e executar a linguagem de programação. Na Figura 1, a entrada de código está marcada com a numeração 2.1 e a saída gerada pela execução do código com a numeração 2.2.

Os arquivos dos notebooks apresentam uma boa organização do código, facilidades no acesso à ajuda e a documentação das funções e classes das diferentes bibliotecas. Facilita a reutilização de trechos do código, a boa documentação dos arquivos e a criação de relatórios de trabalho.

2.2 Pandas e Geopandas

Algumas bibliotecas científicas do Python são largamente utilizadas com dependências na implementação de outras bibliotecas. Pode-se citar o pacote Numpy, responsável pela algebra linear, cálculo numérico e operações com vetores e matrizes de maneira geral, a biblioteca Matplotlib, com a função de gerar gráficos e diagramas ou o Scipy, que apresenta métodos de otimização e outras funções de computação científica.

Pandas é a principal biblioteca de análise e manipulação de dados do Python. É construída a partir das bibliotecas Numpy e Scipy e compatível com a biblioteca Matplotlib. O pacote Pandas trabalha com um objeto *data frame,* que permite a leitura manipulação e escrita de uma série de formatos de tabelas e arquivos de bancos de dados e apresenta desempenho superior aos programas de planilha eletrônica, principalmente para grandes volumes de dados.

A biblioteca Geopandas amplia as capacidades do pacote Pandas, possibilitando o armazenamento, leitura, manipulação e gravação de dados espaciais. Estas capacidades são obtidas através da implementação de uma coluna para armazenamento de formas geométricas nos objetos *data frame* do Pandas. A biblioteca Fiona é responsável pelas estruturas de dados que descrevem as geometrias e o pacote Pyproj proporciona eficiência da aplicação de sistemas de projeção geográficas e na mudança de um sistema para outro.

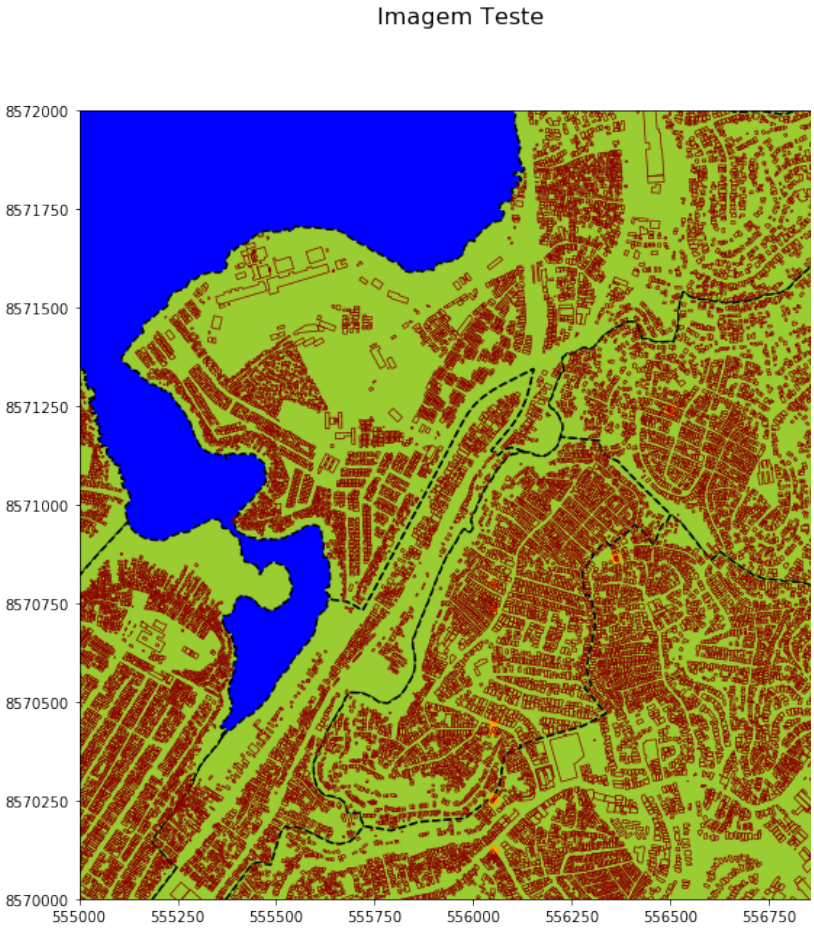

Figura 2: Imagem gerada pela biblioteca geopandas

Com o Pacote Geopandas foi possível ler e escrever arquivos shapefile, gerar mapas, recortar arquivos shapefile, trocar o sistema de coordenadas, gerar estatístic descritiva com eficiência maior que utilizando o QGis e com a vantagem adicional de automatizar o processo.

Na literatura a biblioteca é utilizada por Fissore e Pirotti (2018) na construção de um sistema web para a criação de modelos CityGML a partir de arquivos shapefile.

2.3 OpenStreetMaps e Osmnx

O OpenStreetMap (www.openstreetmap.org) é uma plataforma colaborativa de georreferenciamento e banco de dados online que permite a inserção de dados espaciais por usuários diversos e disponibiliza estas informações para visualização e acesso à informação. O sistema tem por vantagem a possibilidade de se registrar, de maneira colaborativa, as informações geográficas das cidades. A falta de dados é um dos principais empecilhos para o trabalho com GIS e CIM. As desvantagens em se usar o sistema está na falta de controle sobre as inserções de dados, que podem conter erros e imprecisões, dependendo do cuidado no registro das informações por parte de cada usuário colaborador.

Figura 3: Multigrafo de trecho do
sistema viário de Salvador

Testes foram realizados utilizando a biblioteca Omsnx, capaz de ler e baixar dados da plataforma OpenStreetMap. O sistema viário da cidade de Salvador, Ba , Brasil foi utilizado nos testes. Três formas de importação foram testadas: baixando o

arquivo manualmente; importação completa da malha da cidade direto do site, utilizando texto geocoding; importação de um recorte por coordenadas, também diretamente. Os arquivos foram armazenados em objetos *data frame* do Geopandas com sucesso. Após a conversão em *data frame*, foi possível gerar figuras juntando informações importadas dos shapefiles com as baixadas do OpenStreetmap. Pequenas inconsistências são encontradas entre as duas bases de dados, devido diferenças entre as bases utilizadas para a inserção das linhas, pontos e polígonos criados pelos usuários.

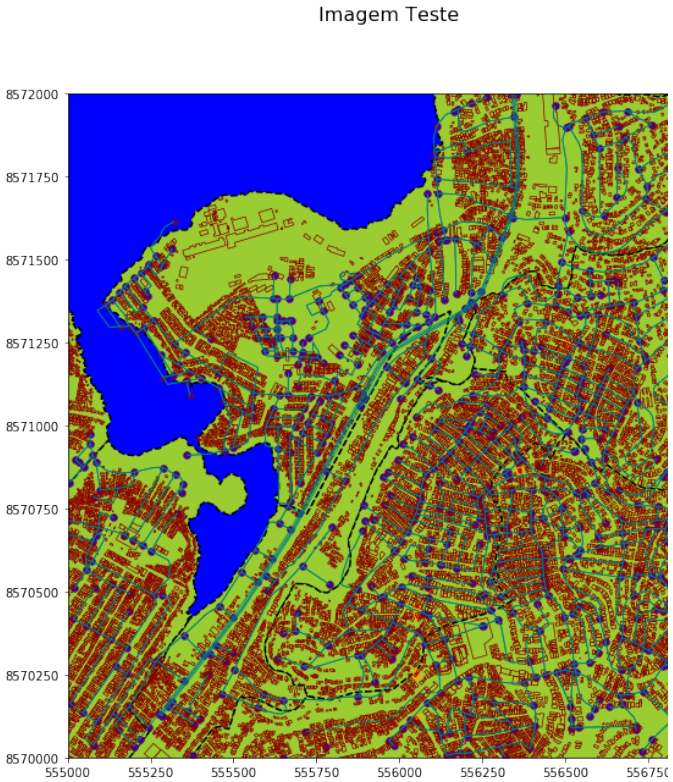

Figura 4: Sobreposição do shapefile com as vias extraidas do OSM

Outro teste realizado foi a tentativa de gravar os dados do sistema viário em formato shapefile. O sistema viário no OpenSteetMap são grafados como um multigrafo, onde os nós são os pontos de encontro entre uma e mais vias e as arestas são o percurso entre estes pontos. A biblioteca Osmnx utiliza métodos do pacote Networx para ler, escrever e operar os multigrafos. O formato shapefile suporta apenas três tipos de variáveis: números inteiros, números reais e texto. As listas de nós, utilizadas para representar os multigrafos, assim como qualquer outro

objeto computacional e as variáveis booleanas não são suportadas e os dados precisam ser transformados para realizar a escrita neste tipo de arquivo.

Os multigrafos são utilizados no estudo matemático das redes complexas, um a subdisciplina da teoria dos grafos. Dentre as principais medidas de uma rede complexa, estão os cálculos de centralidade, sendo as centralidades de grau, proximidade, intermediação e vetor próprio as mais utilizadas. A Figura 5 mostra a centralidade de proximidade da rede viária de salvador.

Como alternativa ao formato shapefile foi testado o armazenamento das informações em um banco de dados geoespacial.

Figura 5: Centralidade de Proximidade da rede complexa do sistema viário de Salvador

2.4 Python e PostGis

Bancos de dados são conjuntos de arquivos relacionados entre si, com o intuito de armazenar e promover acesso à grandes conjuntos de informação. Para o trabalho com banco de dados são utilizados sistemas de gerenciamento de banco de dados (SGBD). Testes de leitura e escrita foram realizados com o sistema PostergSQl e a extensão PostGis, que habilita o SGBD para manipulação de dados geoespaciais. Também foi utilizado uma interface gráfica de gerenciamento de dados PGAmdim4. As versões utilizadas e os sites para *download* estão especificadas do link dos anexos.

Para a gravação e leitura dos *data frames* do Geopandas foram utilizadas as bibliotecas Sqlalchemy e Geoalchemy2. A primeira permite a leitura e escrita de dados em SGBD baseados em SQL e a segunda trabalha com os dados geoespaciais. O processo de escrita e leitura para o banco de dados utilizando estas bibliotecas é bastante eficiente e suportou todos os dados testados, inclusive os que não são aceitos no formato shapefile.

3 CONCLUSÃO

O conjunto de ferramentas cujos testes foram descritos neste relatório se mostraram eficientes e, em muitos aspectos com desempenho, capacidade de automação e replicação em outras bases, superiores às plataformas Gis convencionais. Muitos funções como a leitura e escrita de arquivos dxf, manipulação de arquivos raster (através da biblioteca Rasterio) merecem testes específicos. Quanto ao SGBD utilizado, mais testes devem ser realizados com a adição do 3DCityDB, que permite a importação e exportação de modelos CityGML.

Para a criação e manipulação de modelos CIM é necessário obter informações de diferentes formatos e origens de maneira coordenada e compatível. As ferramentas testadas possuem funcionalidades e interoperabilidade que podem agregar valor na criação e manipulação destes modelos.

Anexos

https://255ribeiro.github.io/arqb30_trabalho_pratico/

REFERÊNCIAS

AMORIM, A. L. DE. Discutindo City Information Modeling (Cim) E Conceitos Correlatos. **Gestão & Tecnologia de Projetos**, v. 10, n. 2, p. 87, 2015.

BERMUDEZ-EDO, M.; BARNAGHI, P.; MOESSNER, K. Analysing real world data streams with spatio-temporal correlations: Entropy vs. Pearson correlation. **Automation in Construction**, v. 88, n. May 2017, p. 87–100, 2018.

BURNING GLASS TECHNOLOGIES. Beyond Point and Click the Expanding Demand for Coding Skills. n. June, p. 11, 2016.

EMC EDUCATION SERVICE. **Data Science and Big Data Analytics: Discovering, Analyzing, Visualizing and Presenting Data**. 1. ed. Indianapolis, Indiana, USA: JOHN WILEY \& SONS, 2015.

FISSORE, F.; PIROTTI, F. **MIGRATION OF DIGITAL CARTOGRAPHY TO CITYGML ; A WEB-BASED TOOL FOR SUPPORTING SIMPLE ETL PROCEDURES**The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences. **Anais**...2018

GIL, J.; ALMEIDA, J.; DUARTE, J. P. The backbone of a City Information Model (CIM). **29th eCAADe conference**, n. Cim, p. 143–151, 2011.

LIM, C.; KIM, K.-J.; MAGLIO, P. P. Smart cities with big data: Reference models, challenges, and considerations. **Cities**, 26 May 2018.

MA, Z.; REN, Y. Integrated Application of BIM and GIS: An Overview. **Procedia Engineering**, v. 196, n. June, p. 1072–1079, 2017.

OSMAN, A. M. S. A novel big data analytics framework for smart cities. **Future Generation Computer Systems**, v. 91, p. 620–633, 2019.

RAMALHO, L. **Python fluente: Programação clara, concisa e eficaz**. [s.l.] NOVATEC, 2015.

TAH, J. H. M.; OTI, A. H.; ABANDA, F. H. A state-of-the-art review of built environment information modelling (BeIM). **Organization, Technology and Management in Construction: an International Journal**, v. 9, n. 1, p. 1638–1654, 2017.

XU, X. et al. From Building Information Modeling to city Information Modeling. **Journal of Information Technology in Construction (ITcon)**, v. 19, n. December 2013, p. 292–307, 2014.