Should "Python for Engineers" be a Course Taught to Freshmen Engineering Majors in the U.S.A. and Abroad?

Dr. Edris Ebrahimzadeh P.E., Des Moines Area Community College

Dr. Edris Ebrahimzadeh is a registered professional engineer (P.E.) and has a Ph.D. in chemical engineering from Brigham Young University (BYU). Currently, he is an instructor of engineering in Des Moines Area Community College and taught for two years at Utah Valley University and Salt Lake Community College.

Dr. Nick M. Safai, Salt Lake Community College

Dr. Nick M. Safai is an ASEE Fellow. He has been an ASEE officer and member for the past 30 years. He has been the six-time elected as the Program Chair of the ASEE International Division for approximately the past 15 years. Three times as the Program Chair for the Graduate Studies Division of ASEE. Nick has had a major role in development and expansion of the ID division. Under his term as the International Division Program Chair the international division expanded, broadened in topics, and the number of sessions increased from a few technical sessions to over eighteen sessions in the recent years. The ASEE International Division by votes, has recognized Nick's years of service through several awards over the past years. Nick has been the recipient of multiple Service awards (examples: 2013, 2010, 2006, 2004, 1996), Global Engineering Educators award (example: 2007, 2005), Best Paper award (examples: 2016, 2010, 2005, 2004, 1995) and other awards from the International Division for exceptional contribution to the international division of the American Society for Engineering Education. Examples of some Awards from other Professional Organizations: • American Society of Civil Engineers (ASCE): Engineering Educator of the Year Award 2004. • Utah Engineers Council, UEC: Engineering Educator of the Year 2005 award, in recognition of outstanding achievements in the field of engineering and for service to society. • SLC Foundation; Salt lake City, Utah: Teaching Excellence Award 2004 and 2012. * SLCC Faculkty Exemplary Service Award April 2015 and 2016. • American Society of Civil Engineers (ASCE): Chapter faculty Advisor recognition award 2002. • Computational Sciences and Education; recognition for outstanding contributions and for exemplary work in helping the division achieve its goals1998. • Engineering Division; recognition for outstanding contributions and for exemplary work in helping the division achieves its goals 1995. • Science and Humanities; recognition for outstanding contributions and for exemplary work in helping the fields achieve its May 1994. • Math & Physical Sciences; appreciation for academic expertise February 1994.

Academics: Nick Safai received his PhD degree in engineering from the Princeton University, Princeton, New Jersey in 1979. He also did a one year post-doctoral at Princeton University after receiving his degrees from Princeton University. His areas of interest, research topics, and some of the research studies have been; • Multi-Phase Flow through Porous Media • Wave propagation in Filamentary Composite Materials • Vertical and Horizontal Land Deformation in a De-saturating Porous Medium • Stress Concentration in Filamentary Composites with Broken Fibers • Aviation; Developments of New Crashworthiness Evaluation Strategy for Advanced General Aviation • Pattern Recognition of Biological Photomicrographs Using Coherent Optical Techniques Nick also received his four masters; in Aerospace Engineering, Civil Engineering, Operation Research, and Mechanical Engineering all from Princeton University during the years from 1973 through 1976. He received his bachelor's degree in Mechanical engineering, with minor in Mathematics from Michigan State. Nick has served and held positions in Administration (Civil, Chemical, Computer Engineering, Electrical, Environmental, Mechanical, Manufacturing, Bioengineering, Material Science), and as Faculty in the engineering department for the past twenty seven years.

Industry experience: Consulting; since 1987; Had major or partial role in: I) performing research for industry, DOE and NSF, and II) in several oil industry or government (DOE, DOD, and NSF) proposals. Performed various consulting tasks from USA for several oil companies (Jawaby Oil Service Co., WAHA Oil and Oasis Co., London, England). The responsibilities included production planning, forecasting and reservoir maintenance. This production planning and forecasting consisted of history matching and

prediction based on selected drilling. The reservoir maintenance included: water/gas injection and gas lift for selected wells to optimize reservoir production plateau and prolonging well's economic life.

Terra Tek, Inc., Salt Lake City, UT, 1985-1987; Director of Reservoir Engineering; Responsible of conducting research for reservoir engineering projects, multiphase flow, well testing, in situ stress measurements, SCA, hydraulic fracturing and other assigned research programs. In addition, as a group director have been responsible for all management and administrative duties, budgeting, and marketing of the services, codes and products.

Standard oil Co. (Sohio Petroleum Company), San Francisco, California, 1983-85; Senior Reservoir Engineer; Performed various tasks related to Lisburne reservoir project; reservoir simulation (3 phase flow), budgeting, proposal review and recommendation, fund authorizations (AFE) and supporting documents, computer usage forecasting, equipment purchase/lease justification (PC, IBM-XT, Printer, etc.), selection/justification and award of contract to service companies, lease evaluation, economics, reservoir description and modeling, lift curves, pressure maintenance (gas injection analysis, micellar-flooding, and water-flooding), Special Core Analysis (SCA), PVT correlations, petrophysics and water saturation mapping.

Performed reservoir description and modeling, material balance analysis. Recovery factors for the reservoir. Administrative; coordination and organization of 2 and 6 week workplans, 1982 and 1983 annual specific objectives, monthly reports, recommendation of courses and training program for the group. Chevron Oil Company, 1979- 1983; Chevron Overseas Petroleum Inc. (COPI), San Francisco, California 1981-1983. Project Leader/Reservoir Engineer, Conducted reservoir and some production engineering work using the in-house multiphase model/simulators. Evaluation/development, budgeting and planning for international fields; Rio Zulia field – Columbia, Pennington Field – Offshore Nigeria, Valenginan, Grauliegend and Rothliegend Reservoir – Netherlands. Also represented COPI as appropriate when necessary.

Chevron Geo-Sciences Company, Houston, TX, 1979-1980 Reservoir Engineer Applications, Performed reservoir simulation studies, history matching and performance forecasting, water-flooding for additional recovery (Rangeley Field – Colorado, Windalia Field – Australia), steam-flooding performances (Kern River, Bakersfield, California), gas blowdown and injection (Eugene Island Offshore Louisiana) on domestic and foreign fields where Chevron had an interest, using Chevron's CRS3D, SIS and Steam Tube simulator programs.

Chevron Oil Field Research Co. (COFRC), La Habra 1978-1979, California. Research Engineer, Worked with Three-Phase, Three-Dimensional Black Oil Reservoir Simulator, Steam Injection Simulator, Pipeflow #2. Also performed history matching and 20-year production forecast including gas lift and desalination plants for Hanifa Reservoir, Abu Hadriya Field (ARAMCO).

“Python for Engineers”: The First Course of Computing for a General Engineering Curriculum

Edris Ebrahimzadeh [a], Nick Safai [b]

Department of Engineering, Des Moines Area Community College, Ankeny, IA 50023, USA
[b] Engineering Department, Salt Lake Community College, Salt Lake City, UT 84123 USA

Abstract

Most freshmen engineering majors have very little or no background in programming. In the first year of college, they learn the basics of programming, so they can apply their computing skills for future engineering courses. Different schools use different programming languages, such as MATLAB, Visual Basic, C++, and Python for their engineering curriculum. However, Python is the only one that is open source. Additionally, the language versatility, online community of users, and powerful analysis packages such as *Numpy* and *Scipy* have made this free utility one of the most popular programming languages among engineers. Literature contains a vast number of references for Python programming for beginners. However, most of these references are written by practitioners in the field of computer science and focus on programming with Python, which do not fit with a general engineering curriculum. Most engineers are problem-solvers and not programmers. This article proposes “Python for Engineers” as a general course concentrated on numerical methods (and not programming) for all engineering disciplines. The article starts by stating the course contents. Then, the paper focuses on six case studies chosen from core sophomore and junior courses in the civil, mechanical, chemical, electrical, petroleum, and industrial engineering curricula to demonstrate how prior knowledge to a powerful and open source computing software could improve the analytical thinking of all engineering students nationally and abroad.

Keywords: Engineering, Education, Software, Python, Curriculum

1. Introduction

Engineers are not expected to write computer codes from scratch; they are more likely expected to utilize built-in functions that have been already written. Therefore, programming by engineering primarily means assembling existing bits of code into a coherent package that solves the problem. The bit of code is a function that implements a specific task. The details of the code are not important for the user. Instead, understanding the input and output of the code (interference) matters the most [1].

Python is a high-level and general-purpose programming language that was first developed by Guido Van Rossum in 1980 [2]. Python is an interpreted language, which enables the programmer to divide the entire code into cells and execute each cell at a time while improving it. In the context of teaching, Python provides four advantages over MATLAB: (1) Python is a free software that can be used in all colleges worldwide. (2) Python has a clear and intuitive syntax. (3) Python is a general-purpose and object-orientated programming language that supports different coding styles. (4) Python has a small core of commands that makes it an easy programming language for beginners. That is why Python is gaining popularity among engineers. It should be mentioned that Python is an emerging language, which means that it is still being developed and refined [3,4].

The article proposes *Python for Engineers* as a general engineering course offered to all freshmen engineering majors. Rather than programming, the course concentrates on the numerical computation capability of Python. The article comes into two parts. In the first part, the contents of the course are discussed. In the second part, six case studies are brought to demonstrate the effectiveness of Python in the analysis of engineering problems.

It should be mentioned that many of the graduate students in the U.S. have completed their undergraduate degrees overseas such as Middle East (including the authors), Africa, and Far East. Education in those countries is usually subsidized by their governments and as a result, not all the engineering colleges in those countries can afford the high cost of software license. Even engineering students in the U.S. do not have easy access to the software that their colleges have provided them. The students have to be in their campuses and use the computers that have the software installed on them to do their homework or work on their course projects. This would be a stumbling block for offering these courses online. However, Python is a free software that can be easily applied in the developing countries at no cost, and can also be used for domestic engineering students remotely. The case studies we will discuss in this article are global and are not limited to any region or specific curriculum.

2. Course contents

The course material is divided into two sections as follows:

Part I-An Introduction to Basic Python Skills

In this course, students are expected to already know the basics of college algebra, trigonometry, and statistics. This section:

1. Describes how Python is applied in different disciplines of engineering and a standard problem-solving procedure is proposed.
2. Shows how to install Python from available platforms (i.e. Spyder IDE, Jupyter Notebook, and online platforms).
3. Introduces the commonly used libraries of Python in engineering. This section also details the wide variety of built-in functions that are available in each of these libraries.
4. Explains how to formulate problems using matrices in Python.
5. Demonstrates the two- and three-dimensional plotting techniques available in Python.

Part II-Basics of Python Programming

In this section students learn how to program in Python. This section:

1. Introduces the logical and control structures.
2. Discusses how to solve linear and nonlinear system of equations using matrix algebra.
3. Introduces data regression and curve fitting.
4. Explains symbolic mathematics packages for the analytical solution and function plotting.
5. Presents numerical differentiation and integration.
6. Describes how to solve ordinary differential equations.

3. Commonly Used Standard Python Libraries in Engineering

In this section, we briefly discuss those libraries of Python that are widely used in engineering disciplines [567].

Numpy and **Scipy**

`Numpy` is the core library for scientific computing in python. This package contains:

- An efficient multi-dimensional array object
- Sophisticated functions
- Tools for integrating with C/C++ and Fortran code
- Efficient linear and random number capabilities

`Scipy` is a complementary package that is built on `Numpy`, and contains modules for linear algebra, interpolation, optimization and ordinary differential equation solvers.

Sympy

This is a Python library for symbolic computation. `Sympy` provides computer algebra capabilities ranging from basic symbolic arithmetic to calculus, discrete mathematics and quantum physics.

Math

This module provides access to a wide range of mathematical functions defined by C standard. All returns of the module are floating numbers, unless stated otherwise.

Matplotlib

`Matplotlib` is a plotting library that produces publication quality figures in different formats and interactive environments. This package can generate scatterplots, histograms, bar charts, error charts, power spectra, etc. with just a few lines of codes.

Scipy.optimize

This is a package that provides several commonly used optimization algorithms. It contains:

- Constrained and unconstrained minimization of multivariate scalar using a wide range of algorithms
- Global optimization techniques
- Curve fitting and least-square minimization algorithms
- Root finders and scalar univariate function minimizers
- Multivariate equation system solvers using a wide range of algorithms.

Scipy.integrate

This sub-package provides several integration methods including an ordinary differential equation (ODE) integrator.

The remainder of the paper focuses on how the skills learned in this course could be applied in different disciplines of engineering.

4. Case Studies

4.1. Statistical Quality Assurance-Industrial Engineering Curriculum

Engineers frequently come across problems that they must decide between two competing statements about the numerical value of a parameter. This decision-making process is called hypothesis testing. All statistical hypotheses are claims about the population of interest. However, in most practical conditions, the variation of the population (σ^2) is unknown. On the other hand, we do not know whether the Normal distribution can probably model the population. Based on the Central Limit Theorem, if the sample size is sufficiently large ($n > 40$), substituting the variation of the sample (S^2) for the variation of the population (σ^2) results in little error in the test procedure, regardless of the shape of the distribution of parameter.

However, if the sample size is not large enough ($n < 40$), then replacing the variation of the population with variation of the sample would result in large error if the probability distribution of the population in not Normal. In such cases, we can plot the Normal probability plot based on the observations and perform the linear regression to fit the data. If the linear equation fits the data well, then we can claim that the probability distribution of the population is Normal. Case study below shows how Python could be effectively utilized for such hypothesis testing analysis.

Golf Club Design

"The increased availability of light materials with high strength has revolutionized the design and manufacture of golf clubs, particularly drivers. Clubs with hollow heads and very thin faces can result in much longer tee shots, especially for players of modest skills. This is due partly to the "spring-like effect" that the thin face imparts to the ball. Firing a golf ball at the head of the club and measuring the ratio of the outgoing velocity of the ball to the incoming velocity can quantify this spring-like effect. The ratio of velocities is called the coefficient of restitution of the club. An experiment was performed in which 15 drivers produced by a club maker were selected at random and their coefficients of restitution measured. In the experiment the golf balls were fired from an air cannon so that the incoming velocity and spin rate of the ball could be precisely controlled. It is of interest to determine if there is evidence (with $\alpha = 0.05$) to support a claim that the mean coefficient of restitution exceeds 0.82. The observations follow [8]."

0.8411	0.8191	0.8182	0.8125	0.8750
0.8580	0.8532	0.8483	0.8276	0.7983
0.8042	0.8730	0.8282	0.8359	0.8660

In this example, the sample mean ($\bar{x}$) and the sample variance (S^2) can be calculated from the data. However, the population variance (σ^2) is unknown, which does not allow us to find the standard deviation of the probability distribution. Additionally, the sample size is less than 40, but if we could prove that the probability distribution of the population is Normal, then the sample variance could be used instead of the population variance to calculate the standard deviation in the test procedure. To investigate whether the distribution of a parameter is Normal, we need to plot the standardized normal scores ($z-$values) versus the observations and then do the linear regression to measure the goodness of the fit.

Python poses a powerful statistical module, `scipy.stats`, which contains a growing library of statistical functions and a large number of probability distributions. `Norm.ppf` is the standard normal distribution function of this package, which by default uses the mean of zero and the standard deviation of one. The function could be utilized to evaluate the $z-$values of a big set of data. Moreover, the `polyfit` and `polyval` functions of the `numpy` library could be employed to perform the linear regression for the graph of computed $z-$values versus the observations. Table 1 summarizes the calculated z-values. **Error! Reference source not found.** clearly demonstrates that the probability distribution of the coefficient of restitution of data is approximately Normal.

The above example is a typical statistical quality assurance problem that junior industrial engineers would face in such a course.

Table 1. Calculations for construction of a normal probability plot.

i	Observed values (x_i)	z_i
1	0.7983	-1.833
2	0.8042	-1.281
3	0.8125	-0.967
4	0.8182	-0.727
5	0.8191	-0.524
6	0.8276	-0.340
7	0.8282	-0.167
8	0.8359	0
9	0.8411	0.167
10	0.8483	0.340
11	0.8532	0.524
12	0.8580	0.727
13	0.8660	0.967

14	0.8730	1.281
15	0.8750	1.833

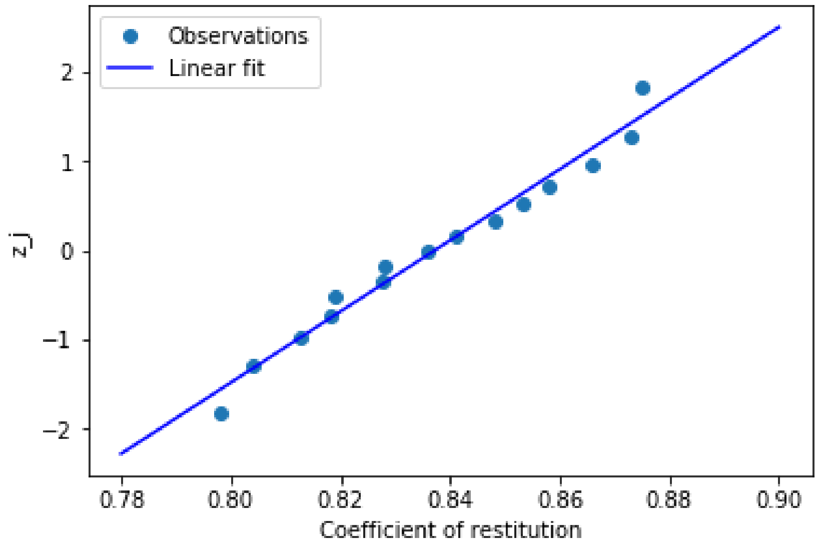

Figure 1. Normal probability plot of the coefficient of restitution data

4.2. Mechanics of Fluids-Civil Engineering Curriculum

Engineers are interested in measuring the resistance to flow in topics varying from nutrient transmission through a plant's vascular system to flow in blood vessels. *Friction factor* (f) is a dimensionless number that measures the resistance of flow in such conduits. Literature contains several empirical equations to compute the friction factor, which *Colebrook equation* is one of the most widely used one:

$$\frac{1}{\sqrt{f}} = -2.0\ \log\left(\frac{\epsilon}{3.7D} + \frac{2.51}{Re\sqrt{f}}\right) \tag{1}$$

where ϵ represents the roughness in (m), D is diameter (m), and Re is the Reynolds number:

$$Re = \frac{\rho V D}{\mu} \tag{2}$$

where ρ is the fluid's density (kg/m^3), V is the fluid's velocity (m/s), and μ is the dynamic viscosity (N.s/m^2). It should be mentioned that Eq. (1) is valid only for $Re > 4000$, which serves as the criterion for turbulent flow. This is an implicit equation for f, which means we cannot find f directly from the equation. To compute f, the solver requires an appropriate initial guess to narrow down the search process. One way of finding the initial guess is to rearrange Eq. (1) and re-write it as $F(f) = 0$:

$$F(f) = \frac{1}{\sqrt{f}} + 2.0\ \log\left(\frac{\epsilon}{3.7D} + \frac{2.51}{Re\,\sqrt{f}}\right) \tag{3}$$

Then function $F(f)$ versus f is plotted to determine the x-intercepts of the graph. Those x −intercepts can serve as the initial guess. The case study below gives more details.

"Air flows through a smooth, thin tube whose $\epsilon = 0.0015$ mm. Other parameters for this problem are $\rho_{air} = 1.23$ kg/m^3, $\mu_{air} = 1.79 \times 10^{-5}$ N.s/m^2, $D = 0.005$ m, and $V_{air} = 40$ m/s. the resistance of air through such a conduit is to be determined [9]."

After making sure the criterion for turbulent floe is satisfied ($Re > 4000$), the function $F(f)$ versus f is plotted using the `sympy.plotting` library of Python, as shown in Figure 2. As can be seen, the x-intercept is in the region of 0.03, so this value is chosen as the initial guess for the solver. Subsequently, the `fsolve` function of the `scipy.optimize` library is applied to find the exact value of f as:

```
f = [0.02896781]
```

The above example is a typical mechanics of fluids problem that sophomore civil engineers would see in such a course.

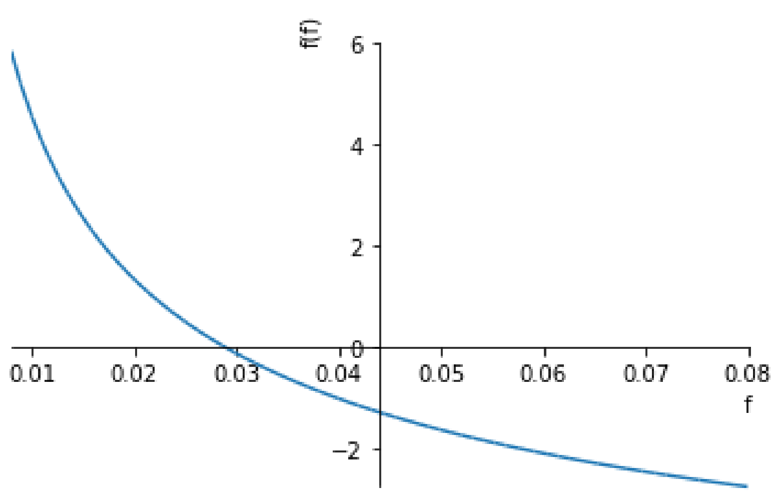

Figure 2. Graph of $F(f)$ versus friction factor

4.3. Engineering Thermodynamics-Mechanical Engineering Curriculum

The first law of thermodynamics, also called energy balance, states that energy can neither be created nor destroyed in a process. Thermodynamic processes involving control volumes are divided into two groups: steady-flow and unsteady-flow (transient-flow) processes. During a steady-flow process, there is no change within the control volume. However, many of the practical engineering process are unsteady-state, which means changes within the control volume with time occur. Therefore, the mass and energy contents of the control volume as well as the energy interactions across the boundary should be considered in the analysis of transient-flow processes. Writing the unsteady-state energy balance for a control volume results in an ordinary differential equation (ODE), which could be easily solved by Python with a few lines of codes. Example below illustrates a quantitative example of such a case.

"An insulated, electrically heated tank for hot water contains190 kg of liquid water at 60 °C when a power outage occurs. If water is withdrawn from the tank at a steady rate of 0.2 kg/s, determine the temperature of the water in the tank after three hours. Assume cold water enters the tank at 10 °C at the same rate it is withdrawn. Also assume heat capacity does not change with temperature and perfect mixing of the water in the tank [10]."

The unsteady-state energy balance in terms of enthalpies, mass, mass flow rate, and internal energy leads to:

$$\frac{d(Mu)}{dt} = \dot{m}_{in}h_{in} - \dot{m}_{out}h_{out} \tag{4}$$

$$\frac{d(Mu)}{dt} = \dot{m}(h_{in} - h_{out}) \tag{5}$$

$$M\frac{du}{dt} = \dot{m}(h_{in} - h_{out}) \tag{6}$$

Water is an incompressible substance and the specific heat capacity at constant volume (C_v) and specific heat capacity at constant pressure (C_p) for incompressible substances are the same. Thus, changes in specific internal energy (u) and specific enthalpy (h) for incompressible substances can be written as:

$$du = CdT \tag{7}$$

$$\Delta h = C(T_{in} - T_{out}) \tag{8}$$

Substituting Eqs. (7) and (8) into the Eq. (6) and applying the assumption of perfect mixing gives:

$$\frac{dT}{dt} = \frac{\dot{m}}{M}(T_{in} - T) \tag{9}$$

where T is the temperature of water inside the tank, T_{in} is the temperature of the incoming water, $\dot{m}$ is the steady flow rate in and out of the tank, and M is the initial mass of water inside the tank. The `scipy.integrate` has the `odeint` function that allows the programmer to easily solve any ordinary differential equation (ODE) over any time interval. Figure 3 illustrates the temperature of water in the tank for three hours.

The above example is a typical engineering thermodynamics problem that sophomore mechanical engineers would come across in such a course.

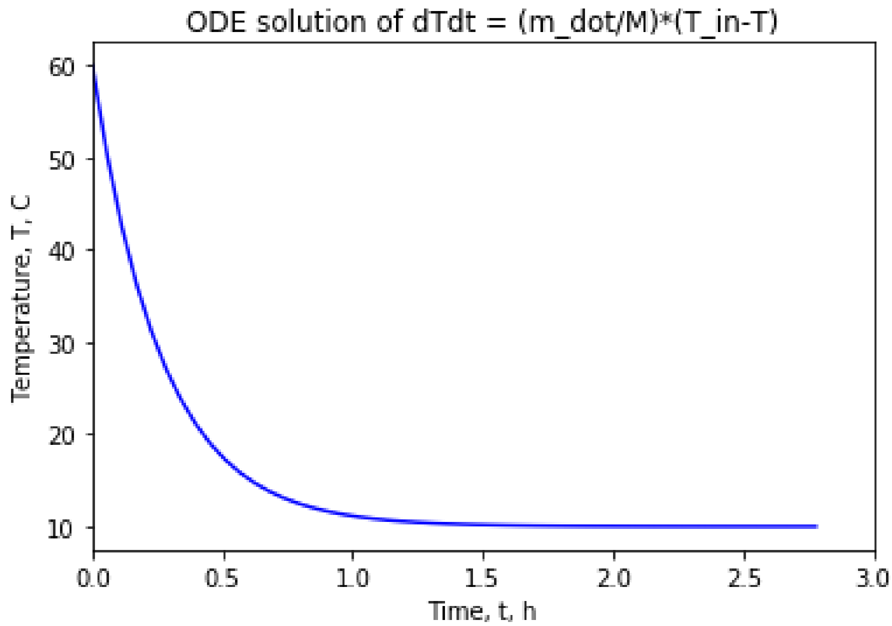

Figure 3. Temperature of water in the tank versus elapsed time.

4.4. Heat Transfer- Petroleum Engineering Curriculum

When electricity moves through any wire, they are being pulled along by an electric field and transfer part of their kinetic energy to the neighboring atoms and raise their temperature. This process is called *Joule heating*. Insulators are used in electrical equipment to transfer the heat generated in wires to the surrounding air. There exists an optimum insulation thickness that would minimize the temperature of wire. Knowing this optimum thickness is essential for the design of wire insulation because any thickness greater than or smaller than this optimum value would result in overheating. Example below demonstrates how Python could assist us in determining the optimum thickness for a wire insulator.

“The steady-state heat flow in a cylindrical shell (Figure 4) is written to compute the steady-state temperature of the wire:

$$T = T_{air} + \frac{q}{2\pi}\left[\frac{1}{k}\ln\left(\frac{r_w + r_i}{r_w}\right) + +\frac{1}{h}.\frac{1}{(r_w + r_i)}\right] \tag{10}$$

where q is the heat generation rate (W/m), r_w is the wire radius (m), r_i is the thickness of insulation (m), k is the thermal conductivity of insulation W/(m.K), and h is convective heat transfer coefficient (W/m^2.K). For a wire with the following parameters, the optimum thickness of the insulator that minimizes the wire's temperature is to be determined [11]."

$q = 75$ W/m, $r_w = 6$ mm, $k = 0.17$ W/(m K), $h = 12$ W/(m^2 K), and $T_{air} = 293$ K.

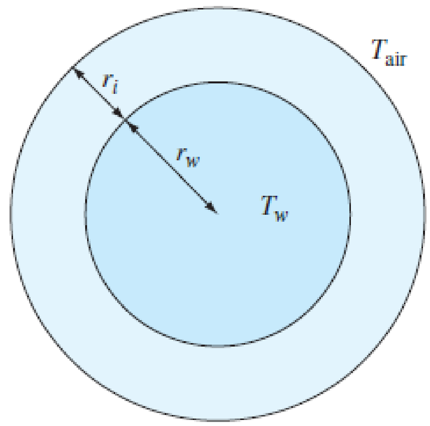

Figure 4. Cross-section of an insulated wire.

The `sympy.plotting` package of Python is utilized to plot the wire's temperature versus the thickness of insulation. Figure 5 clearly shows that the temperature of wire is a parabola that has an absolute minimum value. Using the `fmin` function of the `scipy.optimize` package, the optimum insulation thickness that minimizes the wire's temperature is 8.15 mm. The minimum temperature of wire with this insulation thickness is 423.54 K. The solver successfully terminated the optimization after 11 iterations.

```
Optimization terminated successfully.
         Current function value:
423.539768
         Iterations: 11
         Function evaluations: 22
[8.15625]
```

The above example is a typical heat transfer problem that junior petroleum engineers would come across in such a course.

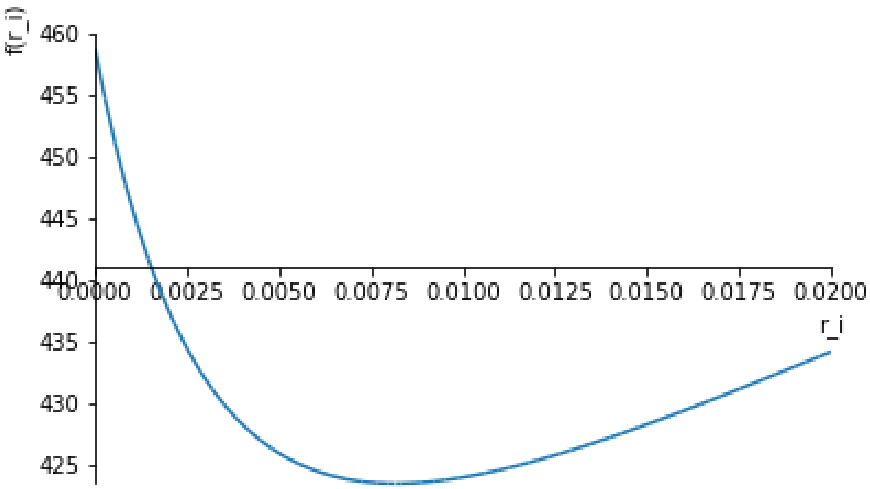

Figure 5. Plot of wire's temperature versus thickness of insulation.

4.5. Chemical Reaction Engineering-Chemical Engineering Curriculum

Chemical reactions are usually characterized by a chemical change, and they yield one or more products. A constituent can be synthesized in a closed system through a sequence of two chemical reactions:

$$2A + B \rightleftarrows C$$

$$A + D \rightleftarrows C$$

The equilibrium constant expression for each chemical reaction is written as:

$$K_1 = \frac{c_c}{c_a^2 c_b} \quad (11)$$

$$K_2 = \frac{c_c}{c_a c_d} \quad (12)$$

where c_i designates the concentration of constituent i. If x_1 and x_2 represent the number of moles of C that are generated in the first and second reactions, respectively, then the concentrations of each constituent can be written in terms of x_1 and x_2 as:

$$c_a = c_{a,0} - 2x_1 - x_2 \quad (13)$$

$$c_b = c_{b,0} - x_1 \quad (14)$$

$$c_c = c_{c,0} + x_1 + x_2 \quad (15)$$

$$c_d = c_{d,0} - x_2 \quad (16)$$

where $c_{i,0}$ represents the initial concentration of each constituent. Substituting these values into Eqs. (11) and (12) gives:

$$K_1 = \frac{(c_{c,0} + x_1 + x_2)}{(c_{a,0} - 2x_1 - x_2)^2(c_{b,0} - x_1)} \quad (17)$$

$$K_2 = \frac{(c_{c,0} + x_1 + x_2)}{(c_{a,0} - 2x_1 - x_2)(c_{d,0} - x_2)} \quad (18)$$

This is a system of non-linear equations with two unknowns. The solution to this problem involves finding the zeros of:

$$F_1(x_1, x_2) = \frac{(c_{c,0} + x_1 + x_2)}{(c_{a,0} - 2x_1 - x_2)^2(c_{b,0} - x_1)} - K_1 \quad (19)$$

$$F_2(x_1, x_2) = \frac{(c_{c,0} + x_1 + x_2)}{(c_{a,0} - 2x_1 - x_2)(c_{d,0} - x_2)} - K_2 \quad (20)$$

"If $c_{a,0} = 50\frac{mol}{m^3}, c_{b,0} = 20\frac{mol}{m^3}, c_{c,0} = 5\frac{mol}{m^3}, c_{d,0} = 10\frac{mol}{m^3}, K_1 = 4 \times 10^{-4}\ m^6.mol^{-2}, K_2 = 3.7 \times 10^{-2}\ m^3.mol^{-1}$, then x_1 and x_2 are to be determined [12]."

Tthe `fsolve` function of the `scipy.optimize` package could be employed to solve this system of non-linear equations. Python returns x_1 and x_2 as:

```
[3.33660129  2.67718089]
```

The above example is a typical chemical reaction engineering problem that junior chemical engineers would come across in such a course.

4.6. Electronic Circuits-Electrical Engineering Curriculum

The current in an AC circuit is usually in the form of a sinusoid wave function because it results in efficient energy transmission:

$$i = i_{peak} \sin(\omega t) \quad (21)$$

$$\omega = \frac{2\pi}{T} \quad (22)$$

where i is the current (A), i_{peak} is the peak current (A), ω is the angular frequency (radians.s^{-1}), and t is time (s). Integration is needed to determine the average current over one period:

$$\bar{\imath} = \frac{1}{T}\int_0^T i_{peak} \sin(\omega t)dt = \frac{i_{peak}}{T}(-\cos(2\pi) + \cos(0))) = 0 \quad (23)$$

The integration leads to zero, which ignores the fact that such a current generates power. To address this issue, electrical engineers determine the root-mean-square current i_{rms}(A) as:

$$i_{rms} = \sqrt{\frac{1}{T}\int_0^T i_{peak}^2 \sin^2(\omega t) dt} = \frac{\sqrt{2}}{2} i_{peak} \tag{24}$$

The above equation shows that i_{rms} is approximately 70% of the peak current for sinusoidal wave function. However, in practical electrical engineering problems, more complicated wave functions (i.e. triangular or square waves) are observed, which makes the analytical integration cumbersome. In such cases, numerical integration methods should be applied to evaluate i_{rms}.

"If the integral that must be evaluated to determine the root-mean-square current of a non-sinusoidal function is [13]"

$$i_{rms}^2 = \int_0^{\frac{1}{2}} (10e^{-t} \sin 2\pi t)^2 \, dt \tag{25}$$

Then the analytical integration would be a very difficult task. However, the `quad` function of `scipy.integrate` library of can be employed to evaluate the integration as:

```
(15.412608048101674, 1.711143232452547e-13)
```

The return value is a tuple, with the first element being the estimated value of the integration and the second element being the upper bound on the error to sixteen significant figures.

To better understand the physical meaning of the estimated i_{rms}, recall that *Ohm's law* states that voltage is directly proportional to current:

$$V = i.R \tag{26}$$

where R is the resistance (Ω). Moreover, *Joule's law* states that instantaneous power absorbed by a circuit element is the product of the current through it and the voltage across it:

$$P = i.V \tag{27}$$

where P is power (W), and V is voltage (V). Substituting Eq. (26) into Eq. (27) gives:

$$P = i^2.R \tag{28}$$

Then, the average power can be computed by integrating Eq. (28) over one cycle with the result:

$$\bar{P} = i_{rms}^2 R \tag{29}$$

This is equivalent to the power generated in a DC circuit with a constant current of i_{rms}.

The above example is a typical electronic circuit's problem that sophomore electrical engineers would see in such a course.

5. Recommendation

The most obvious extension of this work would be to apply the "Python for Engineers" course in the freshmen-engineering curriculum both in the U.S. and abroad, and analyze the data on how the introduction of this new course affected the critical thinking of the engineering students. The most striking advantage of Python over MATLAB and Excel is that it is open source and free. Although almost all engineering colleges in the U.S. are equipped with MATLAB, their students need to be present in their campuses to have access to this software. Note that many engineering colleges in the developing countries cannot even afford Microsoft Office, and therefore they do not have access to Excel. However, these colleges can use Python at no cost.

6. Conclusion

Python is a general-purpose language that is gaining popularity among engineers as well as in the learning environment because:

1. Python is open-source. This is of immense importance in academia because many universities in the developing countries cannot afford the high license fees of other mainstream languages.
2. Python is versatile. Python codes can be written on all major operating systems (i.e. Windows, Mac, OS, Linux, and Unix), and no modifications for the codes are needed to run them in other operating systems.
3. Python is user-friendly and is easy to learn.
4. Python takes advantage of powerful computational packages such as *Numpy* and *Scipy* that facilitate the problem-solving process.

The article proposes *Python for Engineering* as a general engineering course that concentrates on the numerical computation rather than programming itself. The article also contains six case studies chosen from six engineering disciplines to show the impact of Python as a free and powerful computation utility on improving the critical thinking of engineering students nationally and abroad.

7. Acknowledgement

This research was supported by Des Moines Area Community College.

8. References

1. Kiusalaas J. *Numerical Methods in Engineering with Python*. Cabridge University Press; 2010.

2. Rossum V. Python tutorial. Centrum voor Wiskunde en Informatica (CWI), Amsterdam. *www.python.org*. 1995.

3. Fangohr H. A Comparison of C, MATLAB, and Python as Teaching Languages in Engineering. 2004:1210-1217. doi:10.1007/978-3-540-25944-2_157

4. Kavianpour S. The First Course of Programming : Python , Matlab , or C ? The first course of programming : Python or C ? *2016 ASEE Annu Conf Expo*. 2016.

5. Beal L, Hill D, Martin R, Hedengren J. GEKKO Optimization Suite. *Processes*. 2018;6(8):106. doi:10.3390/pr6080106

6. NumPy Reference - Numpy and Scipy Documentation - SciPy.org. https://docs.scipy.org/doc/numpy-1.15.1/reference/.

7. Pyplot Tutorial. https://matplotlib.org/tutorials/introductory/pyplot.html.

8. Douglas C. Montgomery GCR. *Applied Statistics and Probability for Engineers*. Fifth. John Wiley & Sons, Inc.

9. Phillip M. G, Gerhart AL, Hochstein john i. *Fundamentals of Fluid Mechanics*. John Wiley & Sons, Inc.; 2016.

10. Cengel YA, Boles MA. *Thermodynamics: An Engineering Approach*. McGraw-Hill Education; 2014.

11. Incropera FP, DeWitt DP. *Fundamentals of Heat and Mass Transfer*. John Wiley & Sons, Inc.; 2001.

12. Fogler S. *Elements of Chemical Reaction Engineering*. Prentice Hall; 2016.

13. Chapra SC. *Applied Numerical Methods with MATLAB*. McGraw-Hill Education; 2017.