Centre for Modeling and Simulation
Savitribai Phule Pune University
formerly University of Pune

+91.20.2560.1448 • +91.20.2569.0842
cms.unipune.ac.in • office@cms.unipune.ac.in

Master of Technology (M.Tech.) Programme in Modeling and Simulation

Board of Studies: Modeling & Simulation
Faculty: Science & Technology

Savitribai Phule Pune University

May 5, 2018 – Expanded December 7, 2018

About This Document

The *Master of Technology (M.Tech.) Programme in Modeling and Simulation*, designed by a core group of people associated with the Centre for Modeling and Simulation, Savitribai Phule Pune University (formerly University of Pune), was approved by the University in 2007, and came into existence in the academic year 2008-09. Based on the collective and individual experience gained since then, the present document outlines a university-approved revision of this programme.

Citing This Document

Core Curriculum Team and Contributors, *Master of Technology (M.Tech.) Programme in Modeling and Simulation 2018*. Public Document CMS-PD-20180701 of the Centre for Modeling and Simulation, Savitribai Phule Pune University (formerly University of Pune), 2018.
Available at `http://cms.unipune.ac.in/reports`.

Credits and Acknowledgements

Core Curriculum Team: Bhalchandra Pujari, Bhalchandra Gore, Mihir Arjunwadkar.

Contributors: Abhijat Vichare, Vaishali Shah, Bhalchandra Pujari, Padma Pingale, Charulata Patil, Abhay Parvate, V. K. Jayaraman, Bhalchandra Gore, Sukratu Barve, Ashutosh Ashutosh, Mihir Arjunwadkar.

Writing, Collation, Editing: Mihir Arjunwadkar.

Organizational Support: Professors Anjali Kshirsagar and Saroj Ghaskadbi.

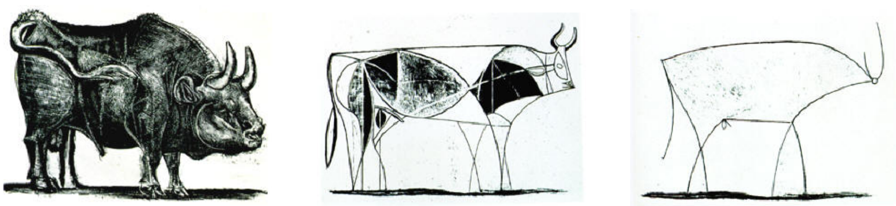
Administrative Support: Mrunalini Dharmadhikari.

We Value Your Feedback

The utility of modeling and simulation as a methodology is extensive, and the community that uses it, academic or otherwise, is diverse. We would appreciate your feedback and suggestions on any aspect of this programme. Feedback can be sent to `office@cms.unipune.ac.in`.

About the Centre

The Centre for Modeling and Simulation, Savitribai Phule Pune University (formerly University of Pune), was established in August 2003 with a vision to promote modeling and simulation methodologies and, in keeping with worldwide trends of modern times, to encourage, facilitate, and support highly interdisciplinary approaches to basic and applied research that transcend traditional boundaries separating individual knowledge disciplines. For more information, visit `http://cms.unipune.ac.in/`.



All models are false, some are useful.

Quote attributed to George E.P. Box.

Contents

Administrative Summary: Master of Technology (M.Tech.) Programme in Modeling and Simulation

Title of the Programme	Master of Technology (M.Tech.) Programme in Modeling and Simulation.
Degree Offered	Master of Technology (M.Tech.) in Modeling and Simulation.
Designed by	Centre for Modeling and Simulation, Savitribai Phule Pune University (formerly University of Pune).
Board of Studies **Faculty**	Modeling and Simulation Faculty of Technology, Savitribai Phule Pune University.
Mode of Operation	Full-time, autonomous programme run by the Centre for Modeling and Simulation, Savitribai Phule Pune University in the academic flexibility/autonomy mode.
Minimum Duration	2 years.
Credits & Breakup	Total credits 100 = 75 core + 25 choice. Semester 1: 21 core + 4 choice. Semester 2: 14 core + 11 choice. Semester 3: 15 core + 10 choice. Semester 4: 25 core.
Structure and Syllabus	This document, Sec. 2 onwards.
Medium of Instruction	English.
Number of Seats	Regular Admissions. 30. Supernumerary Admissions. Sponsored Candidates: up to 10% of the regular seats. Foreign Students, Kashmiri Students, Etc.: as per the prevailing Savitribai Phule Pune University policies.
Fees	Regular Admissions, Kashmiri Students, Foreign Students. As per the prevailing Savitribai Phule Pune University policies for self-supporting departments. Sponsored Candidates. 2 × regular fees for the Maharashtra open category.
Eligibility	{{B.E./B.Tech. any branch} OR {M.Sc.+valid GATE score}} AND {Profficiency in Mathematics at 12+2-level science (i.e., S.Y.B.Sc.) and engineering (i.e., M1+M2+M3) programmes of Savitribai Phule Pune University}
Admissions (First Year)	Regular Admissions Including Kashmiri Students. Performance in an entrance test to ascertain adequate mathematics background equivalent to 12+2-level science (S.Y.B.Sc.) and engineering (M1+M2+M3) programmes of Savitribai Phule Pune University, with a selection threshold/cutoff to be decided by the Centre for Modeling and Simulation and declared in advance. These admissions will be subject to the prevailing Savitribai Phule Pune University admission policy. Sponsored Candidates. First-come-first-served basis; separate entrance test if number of candidates is greater than available seats. Foreign Students. Academic record, statement of purpose, and acceptable English language skills. Telephonic interview and/or letters of recommendation, if necessary, on a case basis.
Admissions (Second Year)	As per the prevailing Savitribai Phule Pune University norms.

1 The Revised M.Tech. Programme

1.1 Overview of This Revision

1. This document details a revision of the 2016 Master of Technology (M.Tech.) Programme in Modeling and Simulation (http://cms.unipune.ac.in/reports/pd-20160121). This document, in particular, lists the changes and additions with respect to the above document, but includes all the syllabi for completeness and convenience.

2. Except for these changes and additions, the basic framework for the Master of Technology (M.Tech.) Programme in Modeling and Simulation as elaborated upon in the above document in Sec. 1 (and in the original version of the programme; namely, http://cms.unipune.ac.in/reports/pd-20070223), remains valid for this revision as well. It has not been reproduced here so as to avoid duplication.

3. The revised curriculum in this document is applicable starting from Academic Year 2018-19 until superseded by the next revision.

Specific changes with respect to the 2016 Master of Technology (M.Tech.) Programme in Modeling and Simulation (http://cms.unipune.ac.in/reports/pd-20160121) are as follows.

1. The proportion of core:choice credits is made exactly 75:25 in conformity with University policies (Sec. 1.2).

2. Prerequisite structure is reviewed extensively and revised (Sec. 1.3).

3. An academically-motivated operational framework for offering courses from this programme as choice-based credit courses to students not affiliated with the Centre for Modeling and Simulation is specified (Sec. 1.4).

4. Based upon past experience and logistic considerations, the Professional Development Programme elaborated upon in the 2016 Master of Technology (M.Tech.) Programme in Modeling and Simulation (Sec. 4) may be instantiated not by default, but depending upon the students' prior exposure to the content of this programme and level of interest.

5. In accordance with University circular 124/2017 (Academic Section) dated 27/5/2017, the eligibility for the programme has now been extended to include M.Sc. with valid GATE score.

1.2 Choice-Based Courses (CBC)

Relative to the 2016 Master of Technology (M.Tech.) Programme in Modeling and Simulation (http://cms.unipune.ac.in/reports/pd-20160121) curriculum, the total number choice credits has been increased from 17 to 25 by transforming the following core courses (total 8 credits) into choice-based courses:

2016 Curriculum, Core Course	CR	This Revision, Choice-Based	Sem.
C107 Computing with R	$1^\dagger$	E013 Computing with R	1
C108 Computing with MATLAB/SciLab	$1^\dagger$	E014 Computing with MATLAB/SciLab	1
C109 Computing with C	$2^\dagger$	E015 Computing with C	1
C202 Transforms	2	E016 Transforms	2, 3
C203 Difference Equations	2	E017 Difference Equations	2, 3

†These 4 choice/elective credits in the first semester are to be used specifically to study one or more relevant programming language/s such as `C`, `Java`, `R`, `Python`, `MATLAB/SciLab`, etc., which is/are supported by readily-available open-source compilers/interpreters/environments.

1.3 Course Prerequisites

1. Prerequisite structure for the entire curriculum has been reviewed and revised.

2. The Common prerequisite (CP) for ALL courses (except courses under the Professional Development Programme, Sec. 4 of the 2016 Master of Technology (M.Tech.) Programme in Modeling and Simulation): Proficiency in Mathematics at 12+2-level science (i.e., S.Y.B.Sc.) and engineering (i.e., M1+M2+M3) programmes of SPPU. Course-specific prerequisites are specified in Sec. 2.1 and 3.1, as well as on individual course pages. Prerequisites need to be expanded recursively to get the full set of prerequisites for a course. Prerequisites for a course are interpreted as indicative of the minimum background necessary to assimilate the course content meaningfully.

1.4 Courses in this Programme Offered as Credit Courses

Courses in this programme (except C401 (Sec. 2.20) Internship, and courses under the Professional Development Programme, Sec. 4 of 2016 Master of Technology (M.Tech.) Programme in Modeling and Simulation) may be offered as credit courses to students affiliated with Savitribai Phule Pune University, education and research institutes/organizations, private universities, etc., as allowed by the provisions of the Savitribai Phule Pune University choice-based credit system. Students wishing to enroll in any of these course/s need approval from

1. their parent department/school/centre,

2. the course instructor/teacher at CMS, and

3. Departmental Committee at CMS.

Fees will be charged as per SPPU rules and policies. Final decision authority on all credit-course enrollment matters will be with CMS Departmental Committee.

2 Core Credits

2.1 Structure of the Core Curriculum

Semester 1

Core credits: 21, choice-based/elective credits: 4†

Code (Sec)	Name	Cr	Prerequisite/s
C101 (Sec. 2.2)	Real Analysis and Calculus	2	CP (Sec. 1.3)
C102 (Sec. 2.3)	Vector Analysis	2	CP (Sec. 1.3)
C103 (Sec. 2.4)	Linear Algebra	2	CP (Sec. 1.3)
C104 (Sec. 2.5)	Ordinary Differential Equations	2	C101 (Sec. 2.2)
C105 (Sec. 2.6)	Partial Differential Equations	3	C104 (Sec. 2.5)
C106 (Sec. 2.7)	Probability Theory	3	CP (Sec. 1.3)
C110 (Sec. 2.8)	Algorithms	2	CP (Sec. 1.3)
C111 (Sec. 2.9)	M&S Hands-On 1	5	CP (Sec. 1.3)

†Choice/elective credits this semester are to be used specifically to study one or more relevant programming language/s such as `C`, `Java`, `R`, `Python`, `MATLAB/SciLab`, etc., which is/are supported by readily-available open-source compilers/interpreters/environments.

Semester 2

Core credits: 14, choice-based/elective credits: 11

Code (Sec)	Name	Cr	Prerequisite/s
C201 (Sec. 2.10)	Complex Analysis	2	C101 (Sec. 2.2)
C204 (Sec. 2.11)	Numerical Computing 1	2	C101 (Sec. 2.2), C102 (Sec. 2.3), C103 (Sec. 2.4), C104 (Sec. 2.5), C105 (Sec. 2.6), C110 (Sec. 2.8), any programming language
C205 (Sec. 2.12)	Optimization 1	2	C101 (Sec. 2.2), C102 (Sec. 2.3), C103 (Sec. 2.4), C104 (Sec. 2.5), C105 (Sec. 2.6), C110 (Sec. 2.8), any programming language
C206 (Sec. 2.13)	Statistical Inference	3	C106 (Sec. 2.7)
C207 (Sec. 2.14)	M&S Hands-On 2	5	C111 (Sec. 2.9)

Semester 3

Core credits: 15, choice-based/elective credits: 10

Code (Sec)	Name	Cr	Prerequisite/s
C301 (Sec. 2.15)	Numerical Computing 2	2	C204 (Sec. 2.11)
C302 (Sec. 2.16)	Optimization 2	3	C205 (Sec. 2.12)
C303 (Sec. 2.17)	Stochastic Simulation	3	C106 (Sec. 2.7)
C304 (Sec. 2.18)	M&S Overview	4	C207 (Sec. 2.14)
C305 (Sec. 2.19)	M&S Apprenticeship	3	C207 (Sec. 2.14)

Semester 4

Core credits: 25, choice-based/elective credits: 0

Code (Sec)	Name	Cr	Prerequisite/s
C401 (Sec. 2.20)	Internship	25	In principle, all prior courses. In practice, internship-dependent, which may be covered under C305 (Sec. 2.19).

2.2 C101 Real Analysis and Calculus

Credits. 2

Prerequisites. CP (Sec. 1.3)

Potentially Dependent Courses. Nearly all courses!

Attributions. Core; C, T

Rationale, Outlook, Purpose, Objectives, and Goals. Ability to apply standard calculus techniques with good comfort levels regarding their basic understanding. The student should also be able to handle real numbers and their analysis as much as outlined in the syllabus. If the student so desires, he/she should be able to build upon this preparation independently to delve into elementary understanding (including proofs and heuristics) as and when needed in later courses.

Syllabus.

1. Basics of set theory, relations and functions.
2. Introduction to metric spaces, open and closed sets, countable sets.
3. Real numbers, real sequences, infinite series, convergence and tests of convergence.
4. Real functions of single and several real variables, plotting graphs of such functions, limits, continuity and uniform continuity.
5. (For real functions of one real variable:) Derivative, Rolle's and Lagrange mean value theorems, Taylor's theorem, order notation and concept of infinitesimal, extreme values and indeterminate forms.
6. (For real functions of several real variables:) Differentiability, Young and Schwarz theorems, partial derivatives, Taylor's theorem and extreme values, homogeneous functions and Euler's theorem, implicit functions, Jacobians.
7. Revision of integration of functions of one variable, definition, standard results and methods of integration, interpretation as area under graph, infinitesimals and Riemann sums.
8. Double integration with procedure and interpretation, Fubini's theorem, change of variables.
9. Triple integration with procedure.

Suggested Texts/References.

1. S. C. Malik and Savita Arora, *Mathematical Analysis.* New Age Publishers, 2009.
2. Erwin Kreyszig, *Advanced Engineering Mathematics.* Wiley India, 2014.

Notes on Pedagogy. Depending upon the capacity of the batch of students, previous orientation and training, the teacher can adjust the depth of delivery so as to best meet the objective. The content can also be tuned accordingly. The content could even be ordered and modified according to the presentation in the prescribed text book/s.

Contributor/s. Sukratu Barve (`http://cms.unipune.ac.in/~sukratu`)

2.3 C102 Vector Analysis

Credits. 2

Prerequisites. CP (Sec. 1.3)

Potentially Dependent Courses. Nearly all courses!

Attributions. Core; C, T

Rationale, Outlook, Purpose, Objectives, and Goals. This foundational course is intended to bring the student at an acceptable level of understanding of vector analysis so that (s)he is able to assimilate related material in advanced courses later on in the programme.

Syllabus.

1. Scalar and vector fields, surfaces and curves in space, examples using analytical geometry, parametric equations for curves and surfaces, intrinsic dimension of subsets of background space using analytical geometry.
2. Continuity and differentiability of vector and scalar fields. Partial derivatives of vectors and scalar fields, the vector operator ∇.
3. Gradient of a scalar field, level surfaces, directional derivative and interpretation of gradient, tangent plane and normal to level surfaces.
4. Divergence and curl expressions. Important vector and scalar calculus identities.
5. Flux of a vector field through a surface portion with an example calculation.
6. Gauss Divergence theorem and outline of proof. Interpretation of divergence in terms of flux.
7. Vector line differential and integral with example calculations.
8. Deriving Green's theorem in a plane. Definition of vector line integral (for 2D vectors on a plane) and relation to Green's theorem.
9. Stokes' theorem and outline of proof (e.g., using Green's theorem). Interpretation of curl in terms of vector line integrals.
10. Conservative vector fields, line integrals and gradients: basic results with proofs, irrotational and solenoidal vector fields.

Suggested Texts/References.

1. Erwin Kreyszig, *Advanced Engineering Mathematics.* Wiley India, 2014.
2. A. R. Vasishtha and Kiran Vasishtha, *Vector Calculus.* Krishna Prakashan Media, 2007.
3. Anil Kumar Sharma, *A Textbook of Vector Calculus.* Discovery Publishing House, 2006.
4. Shanti Narayan and P. K. Mitta, *A Textbook of Vector Calculus.* S. Chand, 1987.

Notes on Pedagogy.

Contributor/s. Sukratu Barve (http://cms.unipune.ac.in/~sukratu)

2.4 C103 Linear Algebra

Credits. 2

Prerequisites. CP (Sec. 1.3)

Potentially Dependent Courses. Nearly all courses!

Attributions. Core; C, T

Rationale, Outlook, Purpose, Objectives, and Goals. This foundational course is intended to bring the student at an acceptable level of understanding of linear algebra so that (s)he is able to assimilate related material in advanced courses later on in the programme.

Syllabus.

1. Introduction to Matrices. Definition and examples, types of matrices, operations on and of matrices (row, column, sum, product, transpose, inverse, Hermitian adjoint) submatrices, determinants, rank, basic theorems on row and column operations on products, theorem on rank of product, elementary matrices, minors, Cofactors, and Cofactor adjoint of a matrix, relation to inverse, standard properties of matrices,symmetry and similarity transformations of matrices.
2. Systems of Linear Equations. Examples, solution methods (Gauss elimination), matrix representation and row echelon form of matrices, basic and free variables, consistency, number of independent equations, Gauss-Jordan Elimination, Cramer's rule and derivation, LU and LDU decomposition.
3. Vector Spaces. Outline of abstract algebra, groups, rings and fields. Definition of vector space over a field and examples of 3D vectors,functions,matrices and their appearance in context of statistics and engineering. Basic results following immediately from axioms. Vector subspaces, linear independence, span, bases, uniqueness of coefficients, dependence theorem, dimension and its uniqueness, direct sums, Transformation of bases.
4. Linear Operators. Definition and properties following immediately from definition. Null space and range. Bases and representation of linear operators as matrices, transformation of operator matrices according to basis transformations, examples.
5. Inner Product Spaces. Definition and basic properties, examples in 3d vectors and spaces of functions, Bessel inequality, Cauchy-Schwarz inequality, Norm from inner product and independent definition of norm. Parallelogram and polarization identities. Angle between vectors and orthogonality (with examples from function spaces). Orthogonal complement of a subset, Orthonormal vectors, their linear independence, Gram-Schmidt Orthogonalization. Optional: Projection operators and orthogonal projection operators, QR decomposition.
6. Eigenvalues, Eigenvectors and Diagonalization. Definition of eigenvectors and eigenvalues of linear operators. Basic results. Calculation using matrix representations. Diagonalization using a particular similarity transformation, application in linear equations. Optional: Linear ODEs, normal matrices and diagonalizability, spectral theorem for normal matrices.
7. Quadratic forms (Optional). Definition, matrix of quadratic form and its symmetrization, definiteness, symmetry transformations and diagonal form, signature, Sylvester's law of inertia, criteria for definiteness, semidefiniteness and indefiniteness. Introduction to higher-degree forms.

Suggested Texts/References.

1. A.K. Lal, *Notes on Linear Algebra*. NPTEL, 2013. http://home.iitk.ac.in/~arlal/book/nptel/pdf/book_linear.pdf
2. Kanti Bhushan Datta, *Matrix and Linear Algebra*. Prentice Hall India, 2008.
3. S. Kumaresan, *Linear Algebra: A Geometric Approach*. Prentice Hall India, 2000.
4. S.K. Mapa, *Higher Algebra: Abstract and Linear*. Levant Books, 2011.
5. Seymour Lipschutz and Marc Lipson, *Linear Algebra (Schaum Series)*. McGraw-Hill India, 2005.
6. Otto Bretscher, *Linear Algebra with Applications*. Pearson, 2008.
7. Paul Halmos and John L. Kelley, *Finite Dimensional Vector Spaces*. Literary Licensing, LLC, 2013.
8. Anil Kumar Sharma, *Linear Algebra*. Discovery Publishing House, 2007.
9. Georgi Shilov, *Introduction to the Theory of Linear Spaces*. Martino Fine Books, 2013.

Notes on Pedagogy.

Contributor/s. Sukratu Barve (http://cms.unipune.ac.in/~sukratu)

2.5 C104 Ordinary Differential Equations

Credits. 2

Prerequisites. C101 (Sec. 2.2)

Potentially Dependent Courses. C105 (Sec. 2.6), C204 (Sec. 2.11), C301 (Sec. 2.15), E002-1 (Sec. 3.4), E002-2 (Sec. 3.5)

Attributions. Core; C, T

Rationale, Outlook, Purpose, Objectives, and Goals. This course, C105 (Sec. 2.6), and E017 (Sec. 3.23) aim at developing commonly-used modeling formalisms for describing change.

Syllabus.

1. Definition of an ordinary differential equation, order and degree along with examples, Definition of solution. General particular and singular solutions, homogeneous functions.
2. First order ODEs having homogeneous functions. Shift of origin change of variables for converting to homogeneous form. Exact first order ODEs. Standard examples of exact ODEs. Integrating factors. Standard examples of integrating factors in various categories of first order ODEs. Linear first order ODEs and integrating factor, Bernoulli's ODE First order ODEs with higher degree and methods of solution (solvable for x, y or dy/dx) Clairaut's form of first order ODE.
3. Second order ODEs (homogeneous and non-homogeneous).
4. ODEs with constant coefficients, (optional examples of analysis of spring-mass-dashpot system) differential operator and its polynomial, complementary and particular integrals, general procedure of obtaining solution.
5. ODEs with variable coefficients, method of variation of parameters (only in case of second order ODEs)
6. Revision of sequences, series and convergence. Series of functions, Power series, ratio test, radius of convergence, series solutions of ODEs, method exemplified in particular cases (second order ODEs) Bessel, Hermite and Legendre ODEs and their series solutions. Brief outline of special functions and their properties.
7. Side conditions of ODEs and their illustration in all the above techniques.
8. Conversion of higher order ODEs into first order ODEs with several dependent variables. Linear ODEs with several dependent variables, matrix formulation, interpretation of characteristic vectors and characteristic values, stability. Applications of this in perturbation of ODEs. Discussion of examples of 6 DOF analysis, control systems stability criteria, predator-prey and chemical reaction ODEs.
9. Numerical methods. Truncation, concept and implementation, forward backward and central difference schemes. Reference to difference equations. Concepts of consistency and stability. Iterative algorithms for solution of difference equations eg. Euler, Heun and Runge Kutta methods. Implicit methods and matrix inversion techniques of solution. Convergence of solutions. Lax theorem (without proof) Computer exercises (3 examples for explicit and 3 for implicit schemes).
10. (OPTIONAL) Existence and uniqueness of solutions of first order ODEs, normed vactor space techniques, uniform continuity, Lifshitz functions and outline of Picard's theorem. Linear ordinary differential operators and resolvents. Examples of resolvents in common

linear ODEs. Relation of side conditions to resolvents. Qualitative analysis of ODEs: Limit sets, fixed points, limit cycles, basins of attractors, Poincare Bendixson theorem (without proof) Lienard's theorem (without proof).

Suggested Texts/References.

1. S. Balachandra Rao and H. R. Anuradha, *Differential Equations with Applications and Programs.* Universities Press, 1996.
2. E. Rukmangadachari, *Differential Equations.* Dorling Kindersley India, 2012.
3. A. Chakrabarti, *Elements of Ordinary Differential Equations and Special Functions.* New Age International, 1990.
4. E. A. Coddington and N. Levinson, *Theory of ordinary Differential Equations.* Tata-McGraw Hill, 1972.
5. G. F. Simmons, *Differential Equations with Applications and Historical Notes.* Tata-McGraw Hill, 1991.
6. G. F. Simmons and S. G. Krantz, *Differential Equations: Theory, Techniques and Practice.* Tata-McGraw Hill, 2007.

Notes on Pedagogy.

Contributor/s. Sukratu Barve (`http://cms.unipune.ac.in/~sukratu`)

2.6 C105 Partial Differential Equations

Credits. 3

Prerequisites. C104 (Sec. 2.5)

Potentially Dependent Courses. C204 (Sec. 2.11), C301 (Sec. 2.15), E002-1 (Sec. 3.4), E002-2 (Sec. 3.5)

Attributions. Core; C, T

Rationale, Outlook, Purpose, Objectives, and Goals. C104 (Sec. 2.5), this course, and E017 (Sec. 3.23) aim at developing commonly-used modeling formalisms for describing change.

Syllabus.

1. Definition of partial differential equation. Order, dependent and independent variables, themes of classification and standard categories, first and second order PDEs; Laplace, heat and wave equations as basic examples of linear second order PDEs. Examples of higher order and non linear PDEs.
2. Cauchy Problems for First Order Hyperbolic Equations. Method of characteristics, Monge cone.
3. Classification of Partial Differential Equations. Normal forms and characteristics for second order PDEs. Principal symbol and quasilinear PDEs, classification of quasilinear PDEs, types of side conditions and principal symbols. General types of side conditions occurring in applications of hyperbolic, parabolic and elliptic PDEs.
4. Initial and Boundary Value Problems. Lagrange-Green's identity and uniqueness by energy methods.
5. (Optional) Stability theory. Energy conservation and dispersion.
6. Laplace equation. Mean value property, weak and strong maximum principle, Green's function, Poisson's formula, Dirichlet's principle, existence of solution using Perron's method (without proof).
7. Heat equation. Initial value problem, fundamental solution, weak and strong maximum principle and uniqueness results (outline of proofs and emphasis on interpretations)
8. Wave equation. Uniqueness, D'Alembert's method, method of spherical means, Duhamel's principle: outline of proofs with emphasis on interpretation.
9. Methods of separation of variables for heat, Laplace and wave equations. Various other methods of solution of PDEs and brief descriptions.
10. Finite difference method as numerical methods for PDEs. Finite Difference Operators, Finite Difference methods, FDM for 1D heat and wave equations, implicit and explicit methods of solution,method of lines,Jacobi, Gauss Seidel and Relaxation methods (for 2D Laplace and Poisson equations) von Neumann stability for difference equations and applications to 2D heat and wave equations. Stability and convergence of matrix difference methods.

Suggested Texts/References.

1. Erich Zauderer, *Partial Differential Equations of Applied Mathematics.* Wiley, 2006.
2. K. Sankara Rao, *Introduction to Partial Differential Equations.* PHI Learning, 2010.

3. Phoolan Prasad and Renuka Ravindran, *Partial Differential Equations.* New Age Publishers, 2012.
4. Lokenath Debnath, *Nonlinear Partial Differential Equations for Scientists and Engineers.* Birkhäuser, 2011.
5. Lawrence C. Evans, *Partial Differential Equations.* American Mathematical Society, 2010.

Notes on Pedagogy.

Contributor/s. Sukratu Barve (http://cms.unipune.ac.in/~sukratu)

2.7 C106 Probability Theory

Credits. 3

Prerequisites. CP (Sec. 1.3)

Potentially Dependent Courses. C206 (Sec. 2.13), C302 (Sec. 2.16), C303 (Sec. 2.17), E003-1 (Sec. 3.7), E003-2 (Sec. 3.8), E007 (Sec. 3.14)

Attributions. Core; C, T

Rationale, Outlook, Purpose, Objectives, and Goals. Probability is the mathematical language for quantifying uncertainty or ignorance, and is the foundation of statistical inference and all probability-based modeling. Goals: Good understanding of probability theory as the basis for understanding statistical inference; familiarity with basic theory and pertinent mathematical results; emphasis on illustrating formal concepts using simulation; and some perspective on modeling using probability by way of real-life contexts and examples.

Syllabus.

1. Probability. Sample spaces and events. Probability on finite sample spaces. Independent events. Conditional probability. Bayes' theorem.
2. Random Variables. Distribution functions and probability functions. Important discrete and continuous random variables. Bivariate and multivariate distributions. Independent random variables. Conditional distributions. Important multivariate distributions. Transformations on one or more random variables.
3. Expectation. Properties. Variance and covariance. Expectation and variance for important random variables. Conditional expectation. Moment generating functions.
4. Inequalities for Probabilities and Expectations. Markov, Chebychev, Hoeffding, Mill, etc. Inequalities for expectation: Cauchy-Schwartz, Jensen, etc.
5. Convergence and Limit Theorems. Notion of convergence for random variables. Types of convergence. Law of large numbers, central limit theorem, the delta method.
6. Stochastic Processes (Optional). Basic introduction to simple branching processes, random walks, Markov chains, etc.

Suggested Texts/References.

1. Christopher R. Genovese, *Working With Random Systems: Mechanics, Meaning, and Modeling*. Unpublished, 2000. `http://www.stat.cmu.edu/~genovese/books/WWRS.ps`
2. Charles M. Grinstead and J. Laurie Snell, *Introduction to Probability*. American Mathematical Society, 1997. `https://math.dartmouth.edu/~prob/prob/prob.pdf`
3. Morris deGroot and Mark Schervish, *Probability and Statistics*. Addison-Wesley, 2002.
4. Larry Wasserman, *All of Statistics*. Springer-Verlag, 2004 (Part 1 of the book).
5. David Stirzacker, *Elementary Probability*. Cambridge University Press, 1994.

Notes on Pedagogy. This course can go hand-in-hand with the *Computing with* `R` course E013 (Sec. 3.19). For example, `R` can be used liberally to illustrate (by the instructor) and explore (by the student) probability-related concepts and important results such as the central limit theorem.

Contributor/s. Mihir Arjunwadkar (`http://cms.unipune.ac.in/~mihir`)

2.8 C110 Algorithms

Credits. 2

Prerequisites. CP (Sec. 1.3)

Potentially Dependent Courses. C204 (Sec. 2.11), C205 (Sec. 2.12), C301 (Sec. 2.15), E005 (Sec. 3.12), E006 (Sec. 3.13), E008 (Sec. 3.15)

Attributions. Core; C, L

Rationale, Outlook, Purpose, Objectives, and Goals. This is intended to be an introduction to algorithms for a non-computer-science graduate student. In principle, any programming language (`C`, `Python`, `Haskel`, `LISP`, etc.) can be used for illustrating algorithms, at the discretion of the instructor. This course is intended for

- familiarizing the student with the computer system organization and its use for problem solving;
- making student understand the need for formal algorithm development; and
- introducing basic types of algorithms, design techniques, data structures used for problem solving.

Syllabus.

1. Introduction to Algorithms. What is an algorithm, why do we need it? Introduction to fundamental algorithms like counting, sorting; algorithms for problem solving using digital computers, flow chart and pseudocode techniques. [5-6 hrs.]
2. Algorithms. Fundamental algorithms and techniques, data structures required (queue, FIFO, FILO, LIFO, LILO terminologies, stacks, link-lists, trees and graphs), logic, set theory, functions, basics of number theory and combinatorics (sequences-series, Sigma and PI notations for termwise summation, multiplication, probability, permutations, combinations), mathematical reasoning–including induction. [10-12 hrs.]
3. Recursion. Need, advantages, disadvantages. Recurrence analysis. Introduction to recurrence equations and their solution techniques (substitution method, tree recursion method, master method). Proof of the master method for solving recurrences. Demonstration of the applicability of master theorem to a few algorithms and their analysis using recurrence equations. Example algorithms: binary search, powering a number, Strassen'S matrix multiplication, etc. [10-12 hrs.]
4. Types of Algorithms and Their Analysis. Theta and big-theta notation, θ and Θ notations; comparison of algorithms, notions of space and time efficiency; as an illustrative example, comparison of quick-sort algorithm with other sorting algorithms can be demonstrated. [5 hrs.]

Suggested Texts/References.

- V. Rajaraman, T. Radhakrishnan, *An Introduction to Digital Computer Design*. PHI, 2007.
- T. H. Cormen, C. E. Leiserson, R. L. Rivest, C. Stein, *Introduction to Algorithms*. PHI Learning, 2009.
- D. E. Knuth, *The Art of Computer Programming, Vol. 1*. Addison Wesley, 2011.

- A. V. Aho, J. E. Hopcroft, J. D. Ullman, *Design and Analysis of Algorithms.* Pearson Education, 2011.
- E. Horowitz, S. Sahni, *Fundamentals of Computer Algorithms.* Universities Press, 2008.

Notes on Pedagogy.

Contributor/s. Bhalchandra Gore (`http://cms.unipune.ac.in/~bwgore`)

2.9 C111 M&S Hands-On 1

Credits. 5

Prerequisites. CP (Sec. 1.3)

Potentially Dependent Courses. C207 (Sec. 2.14), C304 (Sec. 2.18), C305 (Sec. 2.19)

Attributions. Core; C, L, S

Rationale, Outlook, Purpose, Objectives, and Goals. It is a challenge to communicate the depth of the sense in which modeling and simulation are to be understood. This course stems from the belief that a hands on experience can build the intuition more strongly than any other pedagogic technique. It proposes a reasonable cheap laboratory where students build models of some problems, and try to answer questions using them. The models are necessarily physical. The degree of finesse that can be achieved is as much a function of creativity as it is of the cost. The kind of equipment available could start as simply as card boards, pins, glue, paper, colours for painting, wires, bread boards, small motors, or even waste material (e.g. broken toys, devices) etc.

This course is conceived as a first course, and hence has no "syllabus" as such. Instead it is a collection of some "simple"/"simplified" problems that should illustrate the nature of M&S. The key to the success is in grasp on M&S that a student achieves, and should answer basic questions like:

1. What aspect/s of the reality does the model capture? What aspect/s does it **not** capture?
2. What questions can the model answer and what questions can it not answer? Why?
3. Is the model capable of simulation?
4. What questions need a simulation using the model to be answered? Are they different from questions that the model can answer without simulation? If so, in what way?
5. ...
6. Get introduced to M&S through actual experience.
7. What tools (mathematical, statistical, programmatic) are required to address problems in a particular stream?

Syllabus. This is an open-ended course, and the instructor is the best judge of topics and case studies to use to convey the spirit of M&S. At the discretion of the instructor, the selection case studies may include:

1. Study of internal combustion engines
2. Study of a plant cell or animal cell
3. Study of planetary motion of our solar system
4. Study of Newton's Laws of motion (various: e.g. central force, projectiles etc.)
5. Study of equilibrium in chemical reactions
6. Study of mechanical adding machines

Suggested Texts/References. No specific texts or references. Instructor can choose any appropriate selection of texts and references.

Notes on Pedagogy. The main thrust of this course should be to make students comfortable in applying their current knowledge of the modeling techniques to solve a variety of problems. The course may be run by assigning mini-projects to groups of students to generate physical, mathematical, programmatic models and demonstrate their usefulness/inadequacies. The deliverable could be a physical model, a computer program (which is expected to follow basic software development norms) or a proposal based on their study, etc. The exact nature of the deliverable by students and the evaluation methodology is left to the instructor. Hence, there are no prescribed reference books/articles. This course is also a placeholder for the top-down approach to M&S. The students, through case studies, are supposed to understand the need for more detailed study of mathematical, statistical and programmatic tools to understand intricacies of M&S.

Contributor/s. Abhijat Vichare (`https://www.linkedin.com/pub/abhijat-vichare/2/822/828`), Bhalchandra Gore (`http://cms.unipune.ac.in/~bwgore`), Abhay Parvate (`https://jp.linkedin.com/in/abhay-parvate-5b808250`)

2.10 C201 Complex Analysis

Credits. 2

Prerequisites. C101 (Sec. 2.2)

Potentially Dependent Courses. E016 (Sec. 3.22), E001-1 (Sec. 3.2), E001-2 (Sec. 3.3)

Attributions. Core; C, T

Rationale, Outlook, Purpose, Objectives, and Goals. Complex analysis is a powerful and widely used area of mathematics with applicability in diverse areas of science and engineering. This foundational course is intended to bring the student at an acceptable level of understanding of complex analysis so that (s)he is able to assimilate related material in advanced courses later on in the programme.

Syllabus.

1. Complex Analytic Functions. Complex Numbers. Polar form of complex numbers, triangle inequality. Curves and regions in the complex plane. Complex function, limit, continuity, derivative. Analytic function. Cauchy-Riemann equations. Laplace's equation. Rational functions, roots, exponential function, trigonometric and hyperbolic functions, logarithm, general power.
2. Complex Integrals. Line integral in the complex plane. Basic properties of the complex line integral. Cauchy's integral theorem. Evaluation of line integrals by indefinite integration. Cauchy's integral formula. Derivatives of an analytic function.
3. Laurent Series. Review of power series and Taylor Series. Convergence. Uniform convergence. Laurent series, analyticity at infinity, zeros and singularities.
4. Complex Integration by Method of Residues. Analytic functions and singularities. Residues, poles, and essential singularities. The residue theorem. Contours. Contour integration and Cauchy residue theorem as techniques for real integration. Principal values of integrals.

Suggested Texts/References.

1. Tristan Needham, *Visual Complex Analysis.* Oxford University Press, 1999.
2. Erwin Kreyszig, *Advanced Engineering Mathematics.* Wiley India, 2014.
3. M. J. Ablowitz and A. S. Fokas, *Complex Variables: Introduction and Applications.* Cambridge University Press, second edition, 2003.
4. Arfken and Weber, *Mathematical Methods for Physicists.* Elsevier, 2005.

Notes on Pedagogy. On pedagogical note, it is important to remember that students will be required to learn the evaluation of inverse integral transforms later in their course work. It would be useful if the instructor motivates the students using this as application. The student should be adequately familiarized with methods particularly useful in evaluating inverse transforms.

Contributor/s. Bhalchandra Pujari (http://cms.unipune.ac.in/~bspujari)

2.11 C204 Numerical Computing 1

Credits. 2

Prerequisites. C101 (Sec. 2.2), C102 (Sec. 2.3), C103 (Sec. 2.4), C104 (Sec. 2.5), C105 (Sec. 2.6), C110 (Sec. 2.8), any programming language

Potentially Dependent Courses. C301 (Sec. 2.15)

Attributions. Core; C, L

Rationale, Outlook, Purpose, Objectives, and Goals. Many modeling formalisms and situations lead to formulations that involve the use of numerical computing, which we define loosely as numerical analysis/mathematics plus a strong hands-on computing component. This course and its sister course C301 (Sec. 2.15) are intended to cover topics of practical importance that are not covered elsewhere in the curriculum.

Syllabus.

1. Finite-Precision Arithmetic. Computer representations of integers: Properties; overflow and roll-over; endianness. Computer representations of "real" numbers: Fixed-point and floating-point; properties of floating-point numbers and representations (e.g., number and distribution of representable floating-point numbers, overflow and underflow, machine epsilon, minimum and maximum representable number; etc.).
2. The Rounding Error. How elementary arithmetic operations are performed on floating-point numbers. How precision is lost in arithmetic operations such as subtraction. Examples of loss of precision such as in computing roots of a quadratic equation, etc. Analysis of the rounding error for simple computations. Useful strategies for avoiding accumulation of rounding error; e.g., minimize arithmetic operations (e.g., Horner's and other methods for polynomial evaluation), re-arrange computation to avoid subtraction of nearly equal floating-point numbers (e.g., in computing roots of a quadratic equation when $b^2 \approx 4ac$), etc.
3. A User's Perspective on the IEEE 754 Specification. IEEE 754 representations of floating-point types and their characteristics. IEEE 754 specifications for arithmetic operations. Special values: `NaN`, `Inf`, signed zero, etc. Optional: provisions for tracking floating-point exceptions.
4. Roots, Zeros, and Nonlinear Equations in One Variable. Are there any roots anywhere? Examples of root-finding methods. Fixed point iteration, bracketing methods such as bisection, *regula falsi*. Slope methods: Newton-Raphson, Secant. Accelerated Convergence Methods: Aitken's process, Steffensen's and Muller's method.
5. Interpolation. What is interpolation? Polynomial approximation. Polynomial interpolation for derivatives and integration. Newton's form of the interpolation polynomial. The interpolation problem and the vandermonde determinant. The Lagrange form of the interpolation polynomial. Divided differences. The error in polynomial interpolation. Spline interpolation and cubic splines.
6. Approximations. The Minimax approximation problem. Construction of the minimax polynomial. Least-squares and weighted least squares approximations. Solving the least-squares problem: direct and orthogonal polynomial methods.

Suggested Texts/References.

1. David Goldberg, *What Every Computer Scientist Should Know About Floating-Point Numbers*. Computing Surveys, March 1991. http://docs.sun.com/source/806-3568/ncg_goldberg.html
2. Doron Levy, *Introduction to Numerical Analysis*. Unpublished, 2010. http://www.math.umd.edu/~dlevy/books/na.pdf
3. M. T. Heath, *Scientific Computing: An Introductory Survey*. McGraw-Hill, 2002. http://heath.cs.illinois.edu/scicomp/
4. John H. Mathews, *Numerical Methods for Mathematics, Science and Engineering*. Prentice-Hall of India, second edition, 2003.
5. Steven C. Chapra and Raymond P. Canale, *Numerical Methods for Engineers*. Tata McGraw-Hill, third edition, 2000.
6. H. M. Antia, *Numerical Methods for Scientists and Engineers*. Hindusthan Book Agency, second edition, 2002.
7. Kendall E. Atkinson, *An Introduction To Numerical Analysis*. Wiley India, second edition, 2008.

Notes on Pedagogy. This is not intended to be a course on formal numerical analysis per se; the hands-on, computing component needs to be emphasized slightly more, a point-of-view that is consistent with the "concept-over-rigour" viewpoint that is at the heart of this programme. Exercises should involve a mix of paper-and-pencil and computing exercises using any programming language (e.g., `C` together with `GSL`) or computing environment (e.g., `matlab/scilab`, `R`, etc.) that students are familiar with. Modeling contexts in which these numerical methods find their way are left to the discretion of the (expert) instructor.

Contributor/s. Vaishali Shah (https://www.researchgate.net/profile/Vaishali_Shah10)

2.12 C205 Optimization 1

Credits. 2

Prerequisites. C101 (Sec. 2.2), C102 (Sec. 2.3), C103 (Sec. 2.4), C104 (Sec. 2.5), C105 (Sec. 2.6), C110 (Sec. 2.8), any programming language

Potentially Dependent Courses. C302 (Sec. 2.16), E004-1 (Sec. 3.10), E004-2 (Sec. 3.11)

Attributions. Core; C, L

Rationale, Outlook, Purpose, Objectives, and Goals. The ubiquity of optimization formulations in mathematical modeling dictates that a solid background in deterministic optimization methods should be an essential part of a modeler's toolkit.

Syllabus.

1. Preliminaries. Why optimize? A survey of optimization problems and their modeling contexts; system/behaviour $\to$ model or formulation as an optimization problem, the objective function, domain of the objective function, applicable optimization methods. Some terminology (with lots of pictures): a minimizer, local and global minima, constrained and unconstrained minimization, convex minimization. Visualization in 2D and 3D: contours, surfaces, isosurfaces; visualization tools. The optimization interface in the Gnu Scientific Library (`GSL`).
2. One Dimension. Derivatives, conditions for extrema, and the Taylor series. Numerical methods without derivatives: general structure, parabolic interpolation (and its connection with the secant method), golden section search, two-point bracketing and bisection, Golden section search and Brent's method. Numerical methods that use derivative information: Newton-Raphson, Davidon's method, Brent with bracketing. How close to a minimum is numerically close enough? Comparison of, and perspective on, 1D methods.
3. Simplex method.
4. Unconstrained Minimization in More Than One Dimension: Generalities. Partial derivatives and conditions for order-independence. Gradient, Hessian, and directional derivative: properties and interpretation. Taylor expansions in $d = 2$ and in $d = n$. Extrema in N dimensions: necessary conditions for an extremum, extreme values and Taylor expansions, quadratic models, geometry of symmetric bilinear forms, Hessian eigenvalues and eigendirections.
5. Unconstrained Minimization in More Than One Dimension: Methods. Ad hoc methods: a simplex-based method, method of alternating variables. Derivative-free method: Nelder-Mead. Algorithmic structure of commonly-used methods. Steepest descent: rationale, algorithm, and convergence behaviour. Using second-derivative information: Newton method and its convergence behaviour; where and why Newton fails; quasi-Newton methods: rationale and generalities; BFGS and DFP. Direction set methods: basic ideas, Powell's method, conjugate directions and quadratic termination, conjugate gradient method.
6. Constrained Minimization. Equality and bound constraints: general theory. A survey of constrained minimization methods for nonlinear problems.
7. Linear Programming (Optional).
8. Deterministic Global Minimization (Optional). Survey of global minimization problems and deterministic global minimization methods. Comparison with stochastic methods.
9. L_1 Minimization Basics (Optional).

Suggested Texts/References.

1. M. T. Heath, *Scientific Computing: An Introductory Survey*. McGraw-Hill, 2002.. `http://heath.cs.illinois.edu/scicomp/`
2. John H. Mathews, *Numerical Methods for Mathematics Science and Engineering*. Prentice Hall India, second edition, 2003.
3. R. L. Burden and J. D. Faires, *Numerical Analysis*. Brooks Cole, 2004.
4. Forberg, *Numerical Analysis: A practical approach*. McGraw-Hill, 1979.

Notes on Pedagogy. Hands-on work using either `C+GSL` or through computing platforms such as `MATLAB/Scilab` would be beneficial for the students to understand intricacies of optimization problems.

Contributor/s. Vaishali Shah (`https://www.researchgate.net/profile/Vaishali_Shah10`)

2.13 C206 Statistical Inference

Credits. 3

Prerequisites. C106 (Sec. 2.7)

Potentially Dependent Courses. E003-1 (Sec. 3.7), E003-2 (Sec. 3.8), E007 (Sec. 3.14)

Attributions. Core; C, T, L

Rationale, Outlook, Purpose, Objectives, and Goals. Statistical inference is a formalism for reasoning under uncertainty. It is crucial for modeling noisy data, analyzing it, and making inferences from it. In an age where almost every human endeavour is getting data-rich, knowledge of the basics of statistical inference will give an edge to the student. Goals: Good conceptual understanding of the fundamentals of statistical inference, together with the ability to apply them as appropriate; to be able to understand and illustrate formal concepts using simulation.

Syllabus.

1. Overview of Statistical Inference and Learning. Parametric and nonparametric models. Fundamental concepts in inference: point estimation, confidence sets, hypothesis testing.
2. Estimating CDF and Statistical Functionals. The empirical distribution function, properties, confidence band, etc. Statistical functionals. Plug-in estimators for linear statistical functionals.
3. The Bootstrap. Bootstrap variance estimation. Bootstrap confidence intervals.
4. Parametric Inference. Parameter of interest and nuisance parameters. Method of moments (MoM), and properties of MoM estimators. Maximum likelihood (ML) estimation and properties of ML estimators. Multiparameter models. The parametric bootstrap. Role of Assumptions.
5. Hypothesis Testing. Fundamentals of hypothesis testing, type-I and type-II errors, p-values, the Neyman-Pearson lemma, etc. Commonly used tests such as: The Wald test and its connection with confidence interval, Pearson's χ^2 test for multinomial data, the permutation test, the likelihood ratio test, goodness-of-fit tests, t- and F-tests, a few standard tests of normality, correlation test. The multiple testing problem.
6. Bayesian Inference (Optional). The Bayesian philosophy and the Bayesian method. Large sample properties of Bayes procedures. Flat priors, improper priors and "noninformative" priors. Multiparameter problems. Strengths and weaknesses of Bayesian inference *vis a vis* the frequentist/classical approaches.
7. Statistical Decision Theory (Optional). Overview of philosophy, formalism, and methods.

Suggested Texts/References.

- Larry Wasserman, *All of Statistics.* Springer-Verlag, 2004.
- Morris deGroot and Mark Schervish, *Probability and Statistics.* Addison-Wesley, 2002.
- John E. Freund, *Mathematical Statistics.* Prentice-Hall of India, 1998.

Notes on Pedagogy. The emphasis of the course should be on understanding concepts well rather than on mathematical rigour, on being able to interpret formal results and visualize formal constructions, and on being able to apply these concepts and methods to real problems. That said, formal reasoning and analysis should be an integral part of the course wherever it helps understand or illustrate concepts better. The course should also develop a perspective on real-life data modeling contexts where statistical inference plays a crucial role. Hands-on computational work using `R` should be used liberally as a means to illustrate (by the instructor) or understand (by the student) concepts, methods, and applications.

Contributor/s. Mihir Arjunwadkar (`http://cms.unipune.ac.in/~mihir`)

2.14 C207 M&S Hands-On 2

Credits. 5

Prerequisites. C111 (Sec. 2.9)

Potentially Dependent Courses. C304 (Sec. 2.18), C305 (Sec. 2.19)

Attributions. Core; C, L, S

Rationale, Outlook, Purpose, Objectives, and Goals. Same as that for the sister course C111 (Sec. 2.9).

Syllabus. This is an open-ended course, and the instructor is the best judge of topics and case studies to use to convey the spirit of M&S. At the discretion of the instructor, appropriate models could be studied for systems such as:

1. Economic systems (share trading/insurance/banking, etc.)
2. Socio-political systems (social migrations, Bureaucratic Structure and Performance, Electoral Politics and Political Participation, etc.)
3. Water bodies, rains, dams and floods, how the water level would rise/recede, tidal waves, tsunami, etc.
4. Flow of micro-fluidic doses through blood plasma
5. Electrical/Electronic control system, medical instruments
6. Artificial intelligence systems like sound, vision detection, decision making, sensing, etc.
7. Biological systems: evolution, reproduction, self-defence, carbon-cycle related models

Suggested Texts/References. No specific texts or references. Instructor can choose any appropriate selection of texts and references.

Notes on Pedagogy. This course is to be pitched at a level little higher than the sister course C111 (Sec. 2.9). This may be achieved by assigning, say, group projects as part of C111 (Sec. 2.9) and projects for individual or pairs in this course. The aim of this course is to make students realize the nuances of modeling, choice of simulation methods and obtaining useful results through M&S. The students should also be able to take a decision on which tools/subjects should be studied in depth.

Contributor/s. Abhijat Vichare (`https://www.linkedin.com/pub/abhijat-vichare/2/822/828`), Bhalchandra Gore (`http://cms.unipune.ac.in/~bwgore`), Abhay Parvate (`https://jp.linkedin.com/in/abhay-parvate-5b808250`)

2.15 C301 Numerical Computing 2

Credits. 2

Prerequisites. C204 (Sec. 2.11)

Potentially Dependent Courses. None

Attributions. Core; C, L

Rationale, Outlook, Purpose, Objectives, and Goals. Many modeling formalisms and situations lead to formulations that involve the use of numerical computing, which we define loosely as numerical analysis/mathematics plus a strong hands-on computing component. This course and its sister course C204 (Sec. 2.11) are intended to cover topics of practical importance that are not covered elsewhere in the curriculum.

Syllabus.

1. Numerical Differentiation. Basic concepts. Differentiation via interpolation. The method of undetermined coefficients. Numerical derivatives using forward difference, backward difference and central difference. Richardson extrapolation. Differentiation using Lagrange Polynomial, Newton Polynomial.
2. Numerical Integration. Basic concepts. Integration via interpolation. Composite integration rules. Additional integration techniques. The method of undetermined coefficients. Change of an interval. General integration formulas. Simpson integration. The quadrature error. Composite Simpson rule. Gaussian quadrature. Maximizing the quadrature's accuracy. Convergence and error analysis. Romberg integration. Adaptive quadrature basics.
3. Solution of linear systems. Gaussian elimination. Pivoting. Ill-conditioning. Gauss-Jordan method. Matrix inversion. Triangular factorization (LU). Permutation matrices. Cholesky factorization. Iterative methods for linear systems. Diagonally dominant matrices. Jacobi iteration. Gauss-Seidel iteration. Convergence.
4. Eigenvalues and Eigenvectors Homogeneous systems, Power Method, Jacobi's method, Given's, Householder's transformation and Lanczos transformation to tridiagonal form, LR, QL/QR transformation for eigenvalues of tridiagonal matrices, determinants of tridiagonal matrices, Strum sequences, symmetric matrices, band matrices
5. Numerical Linear Algebra Using Software Tools (Optional). Quick working introduction to `BLAS/LAPACK`, and appropriate interfaces in `GSL`, `MATLAB/Scilab`, etc.

Suggested Texts/References.

1. H. M. Antia, *Numerical Methods for Scientists and Engineers*. Hindusthan Book Agency, second edition, 2002.
2. Doron Levy, *Introduction to Numerical Analysis*. Unpublised, 2010.. `http://www.math.umd.edu/~dlevy/books/na.pdf`
3. M. T. Heath, *Scientific Computing: An Introductory Survey*. McGraw-Hill, 2002.. `http://heath.cs.illinois.edu/scicomp/`
4. John H. Mathews, *Numerical Methods for Mathematics, Science and Engineering*. Prentice Hall of India, second edition 2003.

5. D. V. Griffiths and I. M. Smith, *Numerical Methods for Engineers.* Chapman and Hall/CRC, second edition 2011.
6. Steven C. Chapra and Raymond P. Canale, *Numerical Methods for Engineers.* Tata McGraw-Hill, second edition 2000.
7. Curtis F. Gerald and Patrick O. Wheatley, *Applied Numerical Analysis.* Addison-Wesley, fifth edition 1998.
8. Kendall E. Atkinson, *An Introduction To Numerical Analysis.* Wiley India, second edition, 2008.

Notes on Pedagogy. This is not intended to be a course on formal numerical analysis per se; the hands-on, computing component needs to be emphasized slightly more, a point-of-view that is consistent with the "concept-over-rigour" viewpoint that is at the heart of this programme. Exercises should involve a mix of paper-and-pencil and computing exercises using any programming language (e.g., `C`) or computing environment (e.g., `matlab/Scilab`, `R`, etc.) that students are familiar with. Modeling contexts in which these numerical methods find their way are left to the discretion of the (expert) instructor. Coding numerical methods oneself helps most students understand these methods better. The same codes can later on be uses for learning high-performance and parallel computing.

Contributor/s. Vaishali Shah (`https://www.researchgate.net/profile/Vaishali_Shah10`)

2.16 C302 Optimization 2

Credits. 3

Prerequisites. C205 (Sec. 2.12)

Potentially Dependent Courses. None

Attributions. Core; C, L

Rationale, Outlook, Purpose, Objectives, and Goals. Stochastic optimization methods often fare better in situations involving objective functions with multiple local or global minima, when there is combinatorial complexity in the optimization problem, when the goal is to locate a global minimum, and when the measurement or computation of the objective function itself involves uncertainties. A background in these methods should be an essential part of a modeler's toolkit.

Syllabus.

1. Introduction. Formal problem statement. Stochastic vs. deterministic optimization. Principles of stochastic optimization. Local vs. global minimization. An overview of problems involving multiple minima, local or global.
2. Random Search. General properties of direct random search. A few specific algorithms for random search.
3. Stochastic Approximation. Finite-difference SA. Simultaneous perturbation SA.
4. Simulated Annealing. The analogy between optimization and free-energy minimization by a physical system. The travelling salesman problem and SA.
5. Genetic Algorithms. Introduction. Chromosome coding and the basic GA operations. The core genetic algorithm. Implementation aspects. Some perspective on the theory for GAs.
6. More Bio-Inspired Algorithms. Overview of ant-colony and swarm optimization methods.

Suggested Texts/References.

1. James C. Spall, *Stochastic Optimization*, in J. Gentle, W. Härdle, and Y. Mori, eds., *Handbook of Computational Statistics*. Springer, 2004.
2. James C. Spall, *Introduction to Stochastic Search and Optimization: Estimation, Simulation, and Control*. Wiley, 2003.
3. M. Michell, *An Introduction to Genetic Algorithms*. MIT Press, 1996.
4. P. J. M. Van Laarhoven and E. H. L. Aarts, *Simulated Annealing: Theory and Applications*. Kluwer Academic Publishers, 1987.

Notes on Pedagogy. The syllabus outline above is partially based on the first review article above; a qualified instructor experienced in stochastic optimization methods may alter the sequence or topics without altering the overall focus of the course. Methods in a similar vein, such as ant-colony and swarm optimization methods, may be included at the instructor's discretion. This course has some overlap with the concurrently-run course C304 (Sec. 2.18). Ideally, instructors for the two courses should coordinate the delivery of their respective content so that a coherent perspective emerges in the end.

Contributor/s. Mihir Arjunwadkar (`http://cms.unipune.ac.in/~mihir`)

2.17 C303 Stochastic Simulation

Credits. 3

Prerequisites. C106 (Sec. 2.7)

Potentially Dependent Courses. None

Attributions. Core; C, T, L

Rationale, Outlook, Purpose, Objectives, and Goals. To familiarize the student with simulation methods (together with their modeling contexts) involving the use of randomness, including methods for sampling from probability distributions, Monte Carlo integration, Markov chain Monte Carlo methods, etc.

Syllabus.

1. Randomness. Randomness in natural processes: decaying nuclei, chaotic oscillators, leaky faucets, cosmic ray showers, etc. Randomness as complexity, non-compressibility of information, unpredictability, ignorance, statistical independence. Randomness as a modeling assumption. Randomness and entropy.
2. Pseudo-Random Number Generators. Generating deterministic sequences of numbers that appear random. Uniform pseudo-random number generators their properties. Breaking correlations via shuffling. Mersenne Twister and other state-of-the-art generators: an overview. Simple transformations from the Uniform. Other distributions as transformations from the Uniform: exponential, Cauchy, Beta, etc. $N(0,1)$ using Box-Müller and other methods. Arbitrary distributions and acceptance-rejection sampling. Testing for randomness: how random is random enough? `DIEHARD` and other test batteries.
3. (Optional) Correlated Random Numbers. Normal random numbers with pre-specified correlations. Nataf transformation.
4. Monte Carlo Integration. Estimating π using a dartboard. Estimating one-dimensional integrals: basic MC integration. Importance sampling for better estimators and tighter errorbars. Deterministic vs. Stochastic: Behaviour of the error as function of the number of dimensions.
5. Sampling and Integration in More Than One Dimension. Markov chains, their properties, and limit theorems. Metropolis, Metropolis-Hastings and Gibbs sampling. Master equation, detailed balance, and why Metropolis-Hastings works. Relationship between Metropolis-Hastings, Metropolis, and Gibbs. Relationship between Metropolis and rejection sampling. A survey of illustrative problems involving high-dimensional distributions, integration/expectation, and simulations. Practical considerations: the adjustable step length parameter, behaviour of Markov chain Monte Carlo methods when the distribution being sampled is multimodal, burn-in or equilibration behaviour, detecting equilibration/convergence of the Markov chain, convergence diagnostics, correlations and error bars on estimates, etc.
6. (Optional) Specialized (M&)S Methods Involving Randomness. Reaction kinetics, epidemiology, and population dynamics: The Gillespie method. Agent-based stochastic models in epidemiology and other fields. Tutorial on stochastic differential equations. Discrete versus continuous, stochastic versus deterministic: What is more appropriate/useful for given problem?

Suggested Texts/References.

1. Charles M. Grinstead and J. Laurie Snell, *Introduction to Probability*. American Mathematical Society, 1997 or later. https://math.dartmouth.edu/~prob/prob/prob.pdf
2. Hoel, Port, and Stone, *Introduction to Stochastic Processes*. Houghton Mifflin, 1972.
3. George Casella and Edward I. George, *Explaining the Gibbs Sampler*, The American Statistician **46**(3) 167–174 (1992).
4. Brooks, Gelman, Jones, and Meng (eds.), *Handbook of Markov Chain Monte Carlo*. Chapman and Hall/CRC, 2011.
5. Liang, Liu, and Carrol, *Advanced Markov Chain Monte Carlo Methods: Learning from Past Samples*. Wiley, 2010.
6. Gilks, Richardson, and Spiegelhalter, *Markov Chain Monte Carlo Methods in Practice*. Chapman and Hall, 1996.

Notes on Pedagogy. This syllabus is based on an outline for a longer course that was refined over several course deliveries by the contributor (see below). Depending on the background and capabilities of the students, this outline may need to be somewhat diluted or intensified – without compromising upon the essential content and the goals for the course. This course needs sufficient level of hands-on activities, and students require adequate computing skills. What is not mentioned explicitly in the syllabus is the modeling contexts in which stochastic methods can be useful. Exposing the student to such modeling contexts is a must. Specific modeling contexts can be chosen by the instructor according to her/his field of specialization.

Contributor/s. Mihir Arjunwadkar (http://cms.unipune.ac.in/~mihir)

2.18 C304 M&S Overview

Credits. 4

Prerequisites. C207 (Sec. 2.14)

Potentially Dependent Courses. C401 (Sec. 2.20)

Attributions. Core; C, L, S

Rationale, Outlook, Purpose, Objectives, and Goals. This course, together with sister courses C111 (Sec. 2.9) and C207 (Sec. 2.14), are expected to strike a balance between generalities and specific examples, between stochastic and deterministic models, and between describing finished work and actual hands-on modeling on the part of a student. The group of courses C111 (Sec. 2.9), C207 (Sec. 2.14), C304 (Sec. 2.18), and C305 (Sec. 2.19) are the heart of this programme, as they attempt to put the diversity of topics in the curriculum in a unified perspective. They are expected to convey the spirit of mathematical modeling in a coherent manner, through a proper mix of discussion of principles with aptly chosen case-studies from diverse fields, thus putting the methodological training of the rest curriculum in a unified perspective.

Syllabus.

- Foundations of M&S. Modeling, Simulation, M&S as a newly evolved discipline of study, The multidisciplinary nature of M&S; Basic concepts, terms and their definitions; Types of models (mathematical, numerical, statistical, physical, finite element, finite volume, finite domain time difference (FDTD), data-based, agent based, etc.); Actual system and its model: what to expect.
- M&S Characteristics and Descriptors. M&S paradigms–continuous, sampled, event-based, etc.; Attributes–sensitivity, constraints, resolution/granularity, etc.; Verification, Validation and Accreditation of models.
- Classification of models. Classifications based on nature of realization of the model–physical, mathematical, statistical, graphical, etc.; classification based on nature of equation, nature of control parameters and output parameters; contrasting pairs like continuous/discrete, linear/non-linear, deterministic/stochastic, real-time/batch, static/dynamic, time varying/steady state, etc.;
- M&S Process Cycle. model phase, code phase, execution phase, analysis phase, testing, verification and validation phase; feedback mechanism for improvements, quality assurance
- Tools Required for M&S Mathematical, Statistical, Numerical, Programmatic; The need to learn all of them; their use and estimating the nature of final outcome
- Case Studies. Various M&S case studies to bring out the connections between the topics learned and their applications may be chosen. Given the diversity of the M&S enterprise, these are expected to be pedagogically most demanding and a challenge to the instructor.

Suggested Texts/References.

- J. A. Sokolowski and C. M. Banks (Ed.), *Modeling and Simulation Fundamentals.* Wiley, 2010.
- B. P. Zeigler, Herbert Praehofer and Tag Gon Kim, *Theory of Modeling and Simulation.* Academic Press India, 2000.

- A. M. Law, *Simulation, Modeling & Analysis.* McGraw-Hill Education, 2014.
- P. Saxena, *Modeling and Simulation.* Narosa, 2014.
- J. N. Kapur, *Mathematical Modelling.* New Age International Publishers, 2015.

Notes on Pedagogy.

Contributor/s. Bhalchandra Gore (`http://cms.unipune.ac.in/~bwgore`), Sukratu Barve (`http://cms.unipune.ac.in/~sukratu`)

2.19 C305 M&S Apprenticeship

Credits. 3

Prerequisites. C207 (Sec. 2.14)

Potentially Dependent Courses. C401 (Sec. 2.20)

Attributions. Core; C, L, S

Rationale, Outlook, Purpose, Objectives, and Goals. This is an individualized course where a student is to work closely with a mentor. The emphasis of this course is on developing problem-solving, self-learning/self-study, and presentation (written and oral) skills. With the help of the mentor, the student should carry out hands-on work related to the broad focus of the programme, and culminating into a written report and a presentation. With their internship C401 (Sec. 2.20) on the horizon, this course is intended to prepare students to develop appropriate skills including but not limited to literature search, resourcefulness, hands-on problem-solving related to a topic not studied before, etc. See pedagogic notes below.

Syllabus. No fixed syllabus. Mentor to decide the best strategy to achieve aims and objectives of this course for each student separately.

Suggested Texts/References. No prescribed texts. The mentor and the student can use any text or reference individually depending on the topic of study.

Notes on Pedagogy. The role of the mentors is crucial for the success of this course. If a student has already decided her/his place of internship C401 (Sec. 2.20), advisor, topic, etc., then the mentor should make sure, in collaboration with the organization or advisor for the internship, that (s)he spends her/his time in developing skills and background necessary for the internship. For other students, and until the details of their internship are not finalized, mentor should work on developing students' self-study, presentation (written and oral), and any other skills that may not be covered adequately elsewhere during the programme, or those in which the student may not be adequately trained. Assigning a modeling project can be one possible strategy to achieve the goals of this course. The mentor and the students need to meet on regular basis to ensure good and regular progress.

Contributor/s. Bhalchandra Pujari (`http://cms.unipune.ac.in/~bspujari`), Mihir Arjunwadkar (`http://cms.unipune.ac.in/~mihir`)

2.20 C401 Internship

Credits. 25

Prerequisites. In principle, all prior courses. In practice, internship-dependent, which may be covered under C305 (Sec. 2.19).

Potentially Dependent Courses. None

Attributions. Core; L, S

Rationale, Outlook, Purpose, Objectives, and Goals. Internship is the pinnacle of the Master of Technology (M.Tech.) Programme in Modeling and Simulation. The purpose of the internship is for the students to get in-depth exposure and experience in addressing challenging, real-life problems through M&S methods.

Syllabus. No fixed syllabus. Internship advisor/s and internal mentor/s at the Centre to decide the best strategy to achieve the aims and objectives of the internship for each student separately.

Suggested Texts/References. No prescribed texts.

Notes on Pedagogy. Internships are intended to be individual, and spanning a complete semester. In the best interest of the student, internships in settings external to the Centre (such as industry, research or academic institutes, NGOs, etc., depending on the interests of the student) are recommended, although possibilities of in-house internships are not ruled out. The internship topic/project may span any breadth of the M&S enterprise in any problem domain, from translating a domain-specific problem into an appropriate mathematical model, attempting to get analytical insights into the behaviour of the model, to exploring the behaviour of the model through computing/simulation. While the internship may have a substantial computing/coding component, it is not intended to be a pure software/coding project. The Centre's faculty committee is the supreme authority of all matters relating to the approval of internship topics/projects. The M&S context of the topic/project should be well-understood by the student, and should be brought out clearly in the report and presentations. The student, the external advisor/s, and the student's mentor/s at the Centre should ensure this, and have clarity as to where the internship fits into the M&S enterprise. Evaluation for this course is to be done on the basis of (a) regular reporting by the student to the external advisor/s and the internal mentor/s, (b) one or more mid-term presentations, (c) a final presentation, and (d) a final report.

Contributor/s. Charulata Patil (`charulata@cms.unipune.ac.in`), Mihir Arjunwadkar (`http://cms.unipune.ac.in/~mihir`)

3 Choice-Based Credits: In-House Electives

3.1 Structure of the Choice-Based Curriculum

Code (Sec)	Name	Cr	Prerequisite/s	Sem
E001-1 (Sec. 3.2)	Digital Signal and Image Processing 1	5	C101 (Sec. 2.2), C102 (Sec. 2.3), C103 (Sec. 2.4), C104 (Sec. 2.5), E016 (Sec. 3.22) or equivalent, or as defined by the instructor/s	2, 3
E001-2 (Sec. 3.3)	Digital Signal and Image Processing 2	5	E001-1 (Sec. 3.2) or equivalent, or as defined by the instructor/s	2, 3
E002-1 (Sec. 3.4)	Computational Fluid Dynamics 1	5	C105 (Sec. 2.6), or as defined by the instructor/s	2, 3
E002-2 (Sec. 3.5)	Computational Fluid Dynamics 2	5	E002-1 (Sec. 3.4) or equivalent, or as defined by the instructor/s	2, 3
E002-3 (Sec. 3.6)	Computational Fluid Dynamics Laboratory	1	E002-1 (Sec. 3.4), E011 (Sec. 3.18), and/or as defined by the instructor.	2, 3
E003-1 (Sec. 3.7)	Machine Learning 1	5	C206 (Sec. 2.13) or equivalent, or as defined by the instructor/s	2, 3
E003-2 (Sec. 3.8)	Machine Learning 2	5	E003-1 (Sec. 3.7) or equivalent, or as defined by the instructor/s	2, 3
E003-3 (Sec. 3.9)	Machine Learning Laboratory	1	E003-1 (Sec. 3.7), together with E011 (Sec. 3.18) and/or E013 (Sec. 3.19), and/or as defined by the instructor.	2, 3
E004-1 (Sec. 3.10)	Operations Research 1	5	C205 (Sec. 2.12) or equivalent, or as defined by the instructor/s	2, 3
E004-2 (Sec. 3.11)	Operations Research 2	5	E004-1 (Sec. 3.10) or equivalent, or as defined by the instructor/s	2, 3
E005 (Sec. 3.12)	Concurrent Computing	5	C110 (Sec. 2.8) or equivalent and an appropriate programming language, or as defined by the instructor/s	2, 3
E006 (Sec. 3.13)	High-Performance Computing	5	C110 (Sec. 2.8) or equivalent and an appropriate programming language, or as defined by the instructor/s	2, 3
E007 (Sec. 3.14)	Advanced Data Analysis	2	C206 (Sec. 2.13) or equivalent, or as defined by the instructor/s	2, 3
E008 (Sec. 3.15)	Computing with `Java`	2	C110 (Sec. 2.8) or equivalent, or as defined by the instructor/s	2, 3
E009 (Sec. 3.16)	Theory of Computation	5	As defined by the instructor/s	2, 3
E010 (Sec. 3.17)	Functional Programming	2	As defined by the instructor/s	2, 3
E011 (Sec. 3.18)	Computing with `Python`	2	C110 (Sec. 2.8) or equivalent, or as defined by the instructor/s	2, 3
E013 (Sec. 3.19)	Computing with `R`	1	CP (Sec. 1.3)	1
E014 (Sec. 3.20)	Computing with `MATLAB/Scilab`	1	CP (Sec. 1.3)	1
E015 (Sec. 3.21)	Computing with `C`	2	CP (Sec. 1.3)	1
E016 (Sec. 3.22)	Transforms	2	C101 (Sec. 2.2), C102 (Sec. 2.3), C103 (Sec. 2.4), C201 (Sec. 2.10), a programming language	2, 3
E017 (Sec. 3.23)	Difference Equations	2	C101 (Sec. 2.2)	2, 3
E018-1 (Sec. 3.24)	Complex Networks 1	5	C103 (Sec. 2.4), C106 (Sec. 2.7), C110 (Sec. 2.8)	2, 3
E018-2 (Sec. 3.25)	Complex Networks 2	5	E018-1 (Sec. 3.24), C201 (Sec. 2.10)	2, 3
E019-1 (Sec. 3.26)	Astrostatistics 1	5	C206 (Sec. 2.13)	2, 3
E019-2 (Sec. 3.27)	Astrostatistics 2	5	E019-1 (Sec. 3.26)	2, 3
E020 (Sec. 3.28)	Data Visualization	1	As defined by the instructor	2, 3

3.2 E001-1 Digital Signal and Image Processing 1

Credits. 5

Prerequisites. C101 (Sec. 2.2), C102 (Sec. 2.3), C103 (Sec. 2.4), C104 (Sec. 2.5), E016 (Sec. 3.22) or equivalent, or as defined by the instructor/s

Potentially Dependent Courses. E001-2 (Sec. 3.3)

Attributions. CBC; C, T, L, S

Rationale, Outlook, Purpose, Objectives, and Goals. To introduce student to signal modeling and analysis tools, domain-specific formalisms and techniques, and their applications to solve real-life problems in various fields like electronic communication, signal detection, satellite imaging, medical diagnostic signals and images, video analysis, etc. To make student aware of (a) basic theory and mathematical, statistical and algorithmic tools (b) perspective on modeling by way of real-life contexts, examples and applications, and (c) relevant numerical methods. Specific goals of this course are: Understanding the need for signal processing; understanding the correspondence between actual devices, their operations, and their mathematical-statistical representations; learning mathematical, statistical, and algorithmic tools used to improve the quality of a signal and extract useful information from signal; being able to design simple systems for signal processing (like digital filters, digital spectrum analyzer, etc.) using tools such as transforms (Fourier, Z, wavelet), difference equations, pole-zero and frequency plots, mean, median, variance, histogram, probability distributions, etc.; being able to analyze given signal using appropriate tools and infer about quality, content, etc.

Syllabus.

1. Discrete time signals: sequences; representation of signals on orthogonal basis; sampling and quantization; reconstruction of signals; Nyquist's theorem; analog $\rightleftharpoons$ digital signal conversions.
2. Discrete systems: attributes, classifications, analysis of LTI systems, representation of discrete time systems using difference equations, implementation of discrete time systems; correlation of discrete time signals.
3. Z-transform, frequency analysis, discrete Fourier transform (DFT), fast Fourier transform algorithm, frequency response of a system, spectra at output of LTI system; convolution-deconvolution concepts.
4. LTI system as frequency domain filters; filter characterization, inverse systems.
5. Design of FIR and IIR filters; Gaussian, Butterworth, Chebyshev approximations; low-pass, high-pass, notch, bandpass and band-reject filters; effect of quantization on filters–round-off effects.
6. Adaptive filters; power spectrum estimation.
7. Applications of DSP to speech/music and radar/radio-telescope signal processing.

Suggested Texts/References.

1. J. G. Proakis and D. G. Manolakis, *Digital Signal Processing Principles, Algorithms, and Applications.* Pearson, 2012.
2. Oppenheim and Schafer, *Digital Signal Processing.* PHI Learning, 2008.

Notes on Pedagogy. At the discretion of the instructor, `matlab/octave/scilab` can be used for computing. This has the advantage of letting a student focus more on the domain context rather than on programming.

Contributor/s. Bhalchandra Gore (http://cms.unipune.ac.in/~bwgore)

3.3 E001-2 Digital Signal and Image Processing 2

Credits. 5

Prerequisites. E001-1 (Sec. 3.2) or equivalent, or as defined by the instructor/s

Potentially Dependent Courses. None

Attributions. CBC; C, T, L, S

Rationale, Outlook, Purpose, Objectives, and Goals. See purpose/outlook/rationale/goals for the sister course E001-1 (Sec. 3.2). The focus of this follow-up course is more on digital image processing; e.g., electronic communication, satellite imaging, medical diagnostic images, video analysis, image segmentation, etc. Specific goals of this course are: Understanding the need for image processing; understanding the correspondence between actual devices (camera, X-ray, tomographic devices, etc.), their operations, and their mathematical-statistical representations; learning mathematical, statistical, and algorithmic tools used to improve quality of image and extract useful information from image; being able to design simple systems for image processing (like image smoothing, removal of noise, color spectrum analysis etc.) using tools like mean, median, variance, histogram, probability distribution, spatial and temporal filters, etc.; being able to analyze given image using appropriate tools and infer about quality, content, etc.; learning to segment the given image to isolate different objects from it.

Syllabus.

1. What is digital image processing? Its origin, overview of fields of applications, fundamental steps in digital image processing,components of an image processing system.
2. Digital image fundamentals: Human eye and image formation; EM spectrum, image sensing and acquisition, sampling and quantization, basic relationships among pixels-neighbours, connectivity, regions, boundaries, distance measures.
3. Tools used for digital image processing: matrices and vectors, linear vs. non-linear operations, set and logical operators, image transforms, probabilistic methods.
4. Intensity transformations. Basic functions: negation, log, gamma transformation, histogram processing.
5. Spatial filtering: fundamentals, smoothing and sharpening spatial filters, unsharp masking and highboost filtering, use of gradients for non-linear image sharpening, Laplacian operator, using fuzzy techniques for spatial filtering and for intensity transformations
6. Filtering in frequency domain: Fourier series and transform, DFT, FFT, generalization for 2D image space, basic filtering in frequency domain, correspondence between spatial and frequency domain filtering, image smoothing and sharpening using frequency filters, low-pass, high-pass and band-pass filters, notch and band-reject filters, Gaussian, Butterworth, Laplacian filters.
7. Image restoration and reconstruction: Image degradation model, modeling noise, noise reduction using spatial filtering, periodic noise reduction using frequency domain filtering, estimating image degradation function using observation, experimentation and modeling; inverse filtering, statistical (Wiener's mean square error filtering, constrained least square filtering, mean filtering (arithmetic, geometric and harmonic); image restoration from projections (fundamentals of tomography).
8. Colour image processing; colour models: RGB, CMY, CMYK, HSI, relations between them; colour transformations, colour smoothing and sharpening.

9. Wavelets: Haar, Daubechies; DWT, 2D generalization, significance of wavelet coefficients, compression using wavelets.
10. Image compression techniques: spatial and temporal redundancy in images, fidelity criteria, image compression models, some image compression methods (using Huffman, Golomb, arithmetic, LZW, run-length, bit-plane and block-transform coding), relation between block-transform coding and wavelets.
11. Image segmentation fundamentals; point, line and edge detection, thresholding, region-based segmentation.
12. Overview of topics like Video analysis, morphology, watermarking, object/pattern recognition, compressed sensing, etc.

Suggested Texts/References.

1. R. C. Gonzalez and R. E. Woods, *Digital Image Processing, Third Edition.* Pearson, 2013.
2. Bose and Tamal, *Digital Signal and Image Processing.* Wiley India, 2008.

Notes on Pedagogy. At the discretion of the instructor, the computer vision library `OpenCV` can be introduced and used profusely in addition to `matlab/octave/scilab` to ease the programming effort, allowing a student to focus on the formalism and the domain context.

Contributor/s. Bhalchandra Gore (`http://cms.unipune.ac.in/~bwgore`)

3.4 E002-1 Computational Fluid Dynamics 1

Credits. 5

Prerequisites. C105 (Sec. 2.6), or as defined by the instructor/s

Potentially Dependent Courses. E002-2 (Sec. 3.5)

Attributions. CBC; C, T, L, S

Rationale, Outlook, Purpose, Objectives, and Goals. Develop intermediate-level understanding and hands-on skills in the domain of computational fluid dynamics.

Syllabus.

1. Elementary Concepts. Background space, coordinate systems. Fields, scalars, vectors, tensors, transformations, distance metric. Concepts of vector calculus (flux, line integral, Gauss and Stokes theorems). Index notation and Einstein convention. Total derivative, integral curves, velocity field and co-moving derivative.
2. Balance Equations. Equation of continuity. Jacobians and their rates of change. Lagrangian coordinates. Reynolds theorem. Surface forces and traction vector. Cauchy theorem and concept of stress tensor. Cauchy equation of momentum balance. Angular momentum balance equation. Heat flux density, internal energy density, energy balance equation.
3. Constitutive Relations. Introduction. Thermodynamic stimulus and response, rate of response. Darcy's, Fourier's, Ohm's and Fick's laws, Hooke's law, Newton's law of viscosity. Shear, rotation and dilation of velocity field, Navier-Stokes equation, boundary conditions and their importance.
4. Examples of Flow. Hagen-Poisseuille flow, Couette flow and other special cases.

Suggested Texts/References.

1. T. J. Chung, *Computational Fluid Dynamics*. Cambridge University Press, 2002.

Notes on Pedagogy.

Contributor/s. Sukratu Barve (`http://cms.unipune.ac.in/~sukratu`)

3.5 E002-2 Computational Fluid Dynamics 2

Credits. 5

Prerequisites. E002-1 (Sec. 3.4) or equivalent, or as defined by the instructor/s

Potentially Dependent Courses. None

Attributions. CBC; C, T, L, S

Rationale, Outlook, Purpose, Objectives, and Goals. Develop intermediate-level understanding and hands-on skills in the domain of computational fluid dynamics.

Syllabus.

1. FEM Techniques. Finite elements. Shape functions. Finite element interpolation functions. Weighted residual approach. Assembly of element equation. Finite element formulation for advection equation.
2. Finite Volume Approach. Finite volume method. Finite volume discretization. Face area and cell volume. Finite volume via finite difference. Finite volume via finite element method. Comparison of finite difference, finite element, and finite volume methods.
3. Grid Generation. Structured grid generation. Unstructured grid generation. Mesh adaptation. Automatic grid generation for complex geometry problems. Computing techniques.
4. Application to Multiphase Flows.
5. Higher-Order Methods for CFD.
6. Optimization Through CFD. Optimization problem associated with evaluation of first derivative. Optimization problem associated with evaluation of second derivatives.
7. Advanced Fluid Dynamics. Intermediate structures like vortices, boundary layers, shocks, waves and caustics, stream filaments.
8. Numerical Methods. Grid generation techniques for structures and unstructured grids.
9. Hands-On Problem-Solving Through CFD. Implementation of codes for CFD. Computational environments for CFD such as `OpenFOAM, CFDExpert`. `OpenFOAM` architecture, solvers cases and utilities; writing cases and solvers. `CFDEXpert` problems.

Suggested Texts/References.

1. T. J. Chung, *Computational Fluid Dynamics.* Cambridge University Press, 2002.

Notes on Pedagogy.

Contributor/s. Sukratu Barve (`http://cms.unipune.ac.in/~sukratu`)

3.6 E002-3 Computational Fluid Dynamics Laboratory

Credits. 1

Prerequisites. E002-1 (Sec. 3.4), E011 (Sec. 3.18), and/or as defined by the instructor.

Potentially Dependent Courses.

Attributions. CBC; C, L, W

Rationale, Outlook, Purpose, Objectives, and Goals. The purpose of this course is to introduce commonly used (open-source) libraries, tools, packages, and platforms for computational fluid dynamics, so as enhance the industry-preparedness of students. Such tools include, for example, `CFD Python`, `PyCFD`, etc.; meshing tools such as `Gmsh`, etc.; `SWIG`, `PyPar`, `PySPH`, and `SciPy Weave` for development of efficient parallel codes; as also `SU2` and/or `OpenFoam` at a greater depth than what is covered in E002-1 (Sec. 3.4). Not all the tools mentioned above need be covered in the course. The selection of particular tool/s may change from course instance to course instance, depending on (a) the instructor's expertise and preference, (b) industry trends and developments in this field, (c) a balance between depth and breadth, and (d) the number of credits at which the course is being offered, etc.

This course is a flexible-credit choice-based/elective course with the possible number of credits as mentioned above. The number of credits at which it will be offered should be declared well in advance. The depth at which the course is conducted should be decided by the number of credits at which the course is being offered.

Syllabus.

1. A broad survey of commonly used and available tools such as those mentioned above.
2. A crisp introduction to one or two such tools of instructor's choice.
3. Individual or group miniprojects. Students may survey and choose any other appropriate tools, libraries, APIs, etc., best-suited for their chosen topic, under the guidance of the instructor/s.

Suggested Texts/References. Appropriate set of internet and other resources recommended by the instructor.

Notes on Pedagogy. This course should complement the computational fluid dynamics courses E002-1 (Sec. 3.4) and E002-2 (Sec. 3.5). It is intended to be a hands-on course involving case studies and applications, with focus on (open-source) tools. The instructor may consider an open-laboratory, tinkering / experimentation-based approach where different students / groups may explore different (open-source) tools through individual or group mini-projects.

Contributor/s.

1. Mihir Arjunwadkar (`http://cms.unipune.ac.in/~mihir`)
2. Bhalchandra Pujari (`http://cms.unipune.ac.in/~bspujari`)
3. Bhalchandra Gore (`http://cms.unipune.ac.in/~bwgore`)
4. Disha Patil (`https://www.linkedin.com/in/dishapatil/`)

3.7 E003-1 Machine Learning 1

Credits. 5

Prerequisites. C206 (Sec. 2.13) or equivalent, or as defined by the instructor/s

Potentially Dependent Courses. E003-2 (Sec. 3.8)

Attributions. CBC; C, T, L, S

Rationale, Outlook, Purpose, Objectives, and Goals. In the current era of data explosion, scientists and engineers are looking for methods to extract relevant information through automation. Machine learning methods enable development of algorithms to "learn" from the data and make relevant inferences and predictions. This course is designed to introduce students with the subject of machine learning via a variety of statistical tools like classification, clustering, etc. At the end of the course, students are expected to understand the basics of classification and clustering, develop necessary codes, and make correct inferences from given data.

Syllabus.

1. Introduction and Background. What is machine learning? Overview and survey of applications. Problem of induction and statistical inference: Input-output functions, Boolean functions, parametric and non-parametric inference. Probability, certainty, and Bayes theorem. Introduction to typical learning tasks: regression, pattern recognition, feature selection, classification, clustering, rule induction (association). Model validation techniques: cross validation, leave-one-out, majority, etc. Measures of performance of a classifier: confusion matrix, sensitivity, specificity, ROC curves and the AUC, etc.
2. Computational Environments for Machine Learning. Setting up of modeling frameworks (`Weka` and `R`), I/O formats, basic introduction to interfaces.
3. Supervised Learning. Additive models, generative models and discriminative models, logistic regression, Naïve Bayes classifier, linear discriminant analysis, neural networks and support vector machines.
4. Unsupervised Learning fundamentals. Clustering: k-means, hierarchical, self-organizing map. Feature selection via principal component analysis.
5. Laboratory. Models using `Weka` or `R` on UCI benchmark data sets. Writing interfaces for a classifier as derived from a learner. Regression Models. k-means clustering, writing interface for a clusterer.

Suggested Texts/References.

1. T. Hastie, R. Tibshirani, J. H. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer, 2013.
2. Tom Mitchell, *Machine Learning*. McGraw-Hill, 1997.
3. Peter Flach, *Machine Learning: The Art and Science of Algorithms that Make Sense of Data*. Cambridge University Press, 2012.
4. Carl Edward Rasmussen and Christopher K. I. Williams, *Gaussian Processes for Machine Learning*. MIT Press, 2005.
5. Daphne Koller and N. Friedman, *Probabilistic Graphical Models: Principles and Techniques*. MIT Press, 2009.

6. Christopher Bishop, *Pattern Recognition and Machine Learning*. Springer, 2007.
7. Kevin P. Murphy, *Machine Learning: A Probabilistic Perspective*. MIT Press, 2012.
8. Larry Wasserman, *All of Statistics: A Concise Course in Statistical Inference*. Springer, 2004.
9. David MacKay, *Information Theory, Inference and Learning Algorithms*. Cambridge University Press, 2003.
10. Y. S. Abu-Mostafa, Malik Magdon-Ismail, and Hsuan-Tien Lin, *Learning From Data*. AMLBook, 2012.
11. Mehryar Mohri, Afshin Rostamizadeh, and Ameet Talwalkar, *Foundations of Machine Learning*. MIT Press, 2012.

Notes on Pedagogy. Websites such as `http://kaggle.com/` offer challenges based on machine learning. Instructors are encouraged to involve students in such competitions. Organizational, administrative, and infrastructural assistance required for such participation should be provided by the Centre.

Contributor/s. V. K. Jayaraman (`https://in.linkedin.com/in/jayaraman-valadi-a7916925`)

3.8 E003-2 Machine Learning 2

Credits. 5

Prerequisites. E003-1 (Sec. 3.7) or equivalent, or as defined by the instructor/s

Potentially Dependent Courses. None

Attributions. CBC; C, T, L, S

Rationale, Outlook, Purpose, Objectives, and Goals. Students are expected to be familiar with basic tools of machine learning by now. This course should be pitched at slightly advanced level than the first course E003-1 (Sec. 3.7). At the end of this course, students are expected to be able to develop new hybrid algorithms for data analysis and predictions. Students may be introduced to standalone software packages that employ machine learning tools.

Syllabus.

1. Formulation of the Learning Problem. Learning as a statistical problem: estimation of probability measure and basic problems of statistics, learning as density estimation, risk, empirical risk and structural risk, introduction to ill-posed problems and regularization. Learning as an algebraic problem. Learning as a computational problem: learnability, PAC learning, bounds on data, algorithmic learning theory basics. Laboratory: linear models in `R`, writing basic interface for a learner.
2. Reinforcement Learning.
3. Ensemble Methods. Boosting and Bagging.
4. Intelligent Agents.
5. Advanced Clustering Methods. Gaussian/Normal mixture models and the EM algorithm. Fuzzy clustering.
6. Feature Selection. Feature selection using singular value decomposition (SVD). Various filter and wrapper methods.
7. Advanced Applications, Machine Learning Algorithms, and Case Studies. The machine learning approach to time series analysis. Fundamentals and application of clustering/classification to image analysis. Text analytics and natural language processing fundamentals with applications. Social media analytics. Financial analytics.
8. Laboratory: Case studies based on the topics covered. Competitions from `kaggle.com` and similar platforms.

Suggested Texts/References. See the **Suggested Texts/References** section for the sister course E003-1 (Sec. 3.7).

Notes on Pedagogy. Websites such as `kaggle.com` offer rich data sets and challenges for exercising machine learning skills and expertise. Instructors are encouraged to involve students in such competitions. Organizational, administrative, and infrastructural assistance required for such participation should be provided by the Centre.

Contributor/s. V. K. Jayaraman (`https://in.linkedin.com/in/jayaraman-valadi-a7916925`)

3.9 E003-3 Machine Learning Laboratory

Credits. 1

Prerequisites. E003-1 (Sec. 3.7), together with E011 (Sec. 3.18) and/or E013 (Sec. 3.19), and/or as defined by the instructor.

Potentially Dependent Courses.

Attributions. CBC; C, L, W

Rationale, Outlook, Purpose, Objectives, and Goals. The purpose of this course is to introduce commonly used (open-source) libraries, tools, packages, and platforms for machine learning and artificial intelligence, so as enhance the industry-preparedness of students. Such tools include, for example, `TensorFlow`, `Scikit-learn`, `Keras`, `PyTorch`, `Theano`, `GenSim`, `Caffe`, `Chainer`, `Statsmodels`, `Neon`, `Nilearn` in the `Python` world, similar tools in the `R` world, and APIs such as `Apache Spark`, etc. Not all the tools mentioned above need be covered in the course. The selection of particular tool/s may change from course instance to course instance, depending on (a) the instructor's expertise and preference, (b) industry trends and developments in this field, (c) a balance between depth and breadth, and (d) the number of credits at which the course is being offered, etc.

This course is a flexible-credit choice-based/elective course with the possible number of credits as mentioned above. The number of credits at which it will be offered should be declared well in advance. The depth at which the course is conducted should be decided by the number of credits at which the course is being offered.

Syllabus.

1. A broad survey of commonly used and available tools such as those mentioned above.
2. A crisp introduction to one or two such tools of instructor's choice.
3. Individual or group miniprojects. Students may survey and choose any other appropriate tools, libraries, APIs, etc., best-suited for their chosen topic, under the guidance of the instructor/s.

Suggested Texts/References. Appropriate set of internet and other resources recommended by the instructor.

Notes on Pedagogy. This course should complement the machine learning courses E003-1 (Sec. 3.7) and E003-2 (Sec. 3.8). It is intended to be a hands-on course involving case studies and applications, with focus on (open-source) tools. The instructor may consider an open-laboratory, tinkering / experimentation-based approach where different students / groups may explore different (open-source) tools through individual or group mini-projects.

Contributor/s.

1. Mihir Arjunwadkar (`http://cms.unipune.ac.in/~mihir`)
2. Bhalchandra Pujari (`http://cms.unipune.ac.in/~bspujari`)
3. Bhalchandra Gore (`http://cms.unipune.ac.in/~bwgore`)
4. V. K. Jayaraman (`https://in.linkedin.com/in/jayaraman-valadi-a7916925`)

3.10 E004-1 Operations Research 1

Credits. 5

Prerequisites. C205 (Sec. 2.12) or equivalent, or as defined by the instructor/s

Potentially Dependent Courses. E004-2 (Sec. 3.11)

Attributions. CBC; C, T, L, S

Rationale, Outlook, Purpose, Objectives, and Goals.

- To make students aware of the problem domain of operations research (OR).
- To introduce various models and simulation methods in OR.
- Introducing software tools used in OR practices.
- To make students aware of how the methods are used to solve real life problems.

Syllabus.

1. Linear Programming. Simplex algorithm and simplex method, artificial variable methods. Degeneracy, duality in linear programming, duality theorems, dual simplex method with justification. Integer linear programming problem: pure and mixed integer programming problem, Gomory's all integer programming method. Fractional cut method, all integer and mixed integer linear programming problem, branch-and-bound method.
2. Transportation and assignment problems. Balance and degeneracy in transportation and trans-shipment problems. Duality theory of testing optimality of solution in transportation and trans-shipment problems. Hungarian method of assignment. Maximization, prohibition and other variations of assignment problems.
3. Linear programming. Dynamic programming, sensitivity.

   Nonlinear programming. Kuhn-Tucker conditions. Quadratic programming. Wolfes and Beales algorithm for solving quadratic programming problems.
4. Networking models. Network flows. Maximal flow in the network. Transportation, trans-shipment, and assignment problems as networking problems. Network scheduling by PERT/CPM technique. Resource analysis in network scheduling.
5. Introduction to Software Tools. `Lingo` and `AMPL`.

Suggested Texts/References.

1. Kambo, N. S., *Mathematical Programming Techniques.* Affiliated East-West Press, 2002.
2. Taha, H. A., *Operations research.* Pearson, 2014 (9ed).
3. Sierksma, *Linear and Integer Programming.* CRC Press, 2002.
4. Kantiswaroop, P. K., Gupta, M. M., *Operation Research: An Introduction to Management Science.* S. Chand & Co., 2014.
5. Sharma, J. K., *Operations Research Theory and Applications.* Macmillan Publisher India, 2013 (5ed).
6. N. S. Kambo, *Mathematical Programming Techniques.* Affiliated East-West Press, 2002.

Notes on Pedagogy. Each unit in the syllabus above can be covered in approximately 12 laboratory and 3 "theory" sessions.

Contributor/s. Padma Pingle

3.11 E004-2 Operations Research 2

Credits. 5

Prerequisites. E004-1 (Sec. 3.10) or equivalent, or as defined by the instructor/s

Potentially Dependent Courses. None

Attributions. CBC; C, T, L, S

Rationale, Outlook, Purpose, Objectives, and Goals.

- To make students aware of the advanced techniques in Operations Research (OR).
- Introducing various models & theories used in OR.
- To make students able to apply their knowledge for solving real life problems in OR.

Syllabus.

1. Linear Programming Problem. Advanced techniques: Revised Simplex Method, Simplex Method versus Revised Simplex Method, Bounded variable technique, Parametric Linear Programming, Linear Fractional Programming and their applications, Karmarkar algorithm.

   Sequential problems. Basic terms used in sequencing, n-jobs two machines sequencing problems, processing two jobs through K machines sequencing problems.
2. Queuing Theory. Markovian and Non-Markovian queuing models (i.e., $(M/M/1)$, $(M/M/s)$, $(M/E_k/1)$, $(M, G, 1)$, steady-state probabilities and their characteristics, cost profit models of $(M/M/1)$ queuing systems. Simulation, event type simulation, simulation of a queuing system.
3. Inventory Models. Types of inventories. Reasons for carrying inventories, Inventory control, Inventory carrying costs and factors. Concept of economic order quantity (EOQ) or lot size problems. Deterministic inventory problems with and without shortages, EOQ problems with price breaks. Multi-items deterministic problems. Selective inventory control techniques. Inventory problems with uncertain demand, systems of inventory control (Q-system and P-system).
4. Game Theory. Two-person Zero-sum games, Maximin-Minimax principle. Games without saddle point. Graphical solution of $2\times n$ and $m\times 2$ games, Dominance property, Arithmetic methods for $n \times n$ games, General solution of $m \times n$ Rectangular games, Limitations and Extensions.

Suggested Texts/References.

1. Kambo, N. S., *Mathematical Programming Techniques.* Affiliated East-West Press, 2002.
2. Taha, H. A., *Operations research.* Pearson, 2014 (9ed).
3. Sierksma, *Linear and Integer Programming.* CRC Press, 2002.
4. Kantiswaroop, P. K., Gupta, M. M., *Operation Research: An Introduction to Management Science.* S. Chand & Co., 2014.
5. Sharma, J. K., *Operations Research Theory and Applications.* Macmillan Publisher India, 2013 (5ed).

Notes on Pedagogy. Each unit in the syllabus above can be covered in approximately 12 laboratory and 3 "theory" sessions.

Contributor/s. Padma Pingle

3.12 E005 Concurrent Computing

Credits. 5

Prerequisites. C110 (Sec. 2.8) or equivalent and an appropriate programming language, or as defined by the instructor/s

Potentially Dependent Courses. None

Attributions. CBC; C, T, L, S

Rationale, Outlook, Purpose, Objectives, and Goals. Concurrency is ubiquitous, and not only a part of OS courses. It can also be a program modularization technique whereby applications can be organized as a set of interacting concurrent components. The word "concurrency" not only alludes to "occurring at the same time", but also has other connotations like "agreeing on some thing", or "coming together for a task" etc. While concurrency in the first sense is often conflated with parallel computing, concurrency is broader and more pervasive. This course aims to bring out the basic issues and techniques needed to deal with concurrency.

- Understanding of concurrency in parallel and distributed computing
- Designing and modularising programs using concurrent tasks
- Synchronizing and communicating between tasks in a concurrent program
- (Optional): Formal modeling of concurrent systems

Syllabus.

1. Introduction. Introducing concurrency with simple problems like Readers-Writers, Producer-Consumer, Bounded Buffer, etc., as illustrations. Challenges in concurrency: synchronization, mutual exclusion, deadlock, livelock, starvation, non-determinism. Distinction between concurrent and distributed systems. Relation between timing and concurrency. Concurrency in algorithms and physical concurrency. Example of concurrent systems: operating systems, database systems, web servers.
2. Types of concurrency. (a) Program level – Introduction to the variety of control flows: sequential, coroutines etc. Program execution approaches: single tasking, multitasking, multiprocessing, multicomputing and distributed. (b) Data level – data structures and their concurrency, parallel computing.
3. Inter-process communication. Communication between components in a concurrent program: shared memory, message passing. Communication/Data exchange/Interaction between processes on: time shared systems, client-server systems, distributed systems. Components of a concurrent program versus interaction between processes; similarities and differences.

   Techniques of handling concurrency. Locking, Time stamp ordering, Semaphores, Monitors. Goals: Correctness or fault tolerance.
4. Distributed systems. timing and clock synchronization, time stamps, Chandy-Lamport time ordering. Global snapshot on distributed systems. Examples: from distributed snapshots, Synchronous and asynchronous systems.
5. (Sample: See note 1). Introduction to Android as a programming platform. The basic "HelloWorld" application on Android. Implement algorithms on Android – three sort algorithms as independent programs. IPC on Android – A pair of sort algorithms on

shared data set (Scenarios: e.g., (a) One works in place, and the other on a copy – both run concurrently, (b) Both work in place and concurrently, etc. Choose a scenario).

6. (Optional: See note 2). Abstract representation of concurrent (esp. distributed) systems: Introduction to a Process Algebra, e.g., CCS. The Calculus of Communicating Systems: Syntax of the CCS, Operational semantics of the CCS (non-determinism etc.). Worked examples for the CCS.

Suggested Texts/References.

1. Clay Breshears, *The Art of Concurrency: A Thread Monkey's Guide to Writing Parallel Applications*. O'Reilly, 2009.
2. Robin Milner, *Communicating and Mobile Systems: The* π *Calculus*. Cambridge University Press, 1999.
3. Michel Raynal, *Concurrent Programming: Algorithms, Principles, and Foundations*. Springer, 2013.
4. A. Roscoe, *Theory and Practice of Concurrency*. Prentice Hall, 1997.
5. G. Blake Meike, *Programming Android*. Shroff, 2012.
6. Reto Meier, *Professional Android 4 Application Development (Wrox)*. Wiley, 2012.

Notes on Pedagogy.

1. This course requires programming exercises. The proposal illustrates using Android. However, any suitable system could be used based on the familiarity of the students and the instructor. For example, a Unix/Linux based system would bring out shared memory (`shm*`) system calls, message passing (`msg*`) system calls, and possibly simple socket based IPC programming. Yet another possibility is to use MPI based programming assignments with similar detailing. This component is therefore labeled as "Sample" to suggest that the *means* used to bring out concurrency may be decided by the instructor depending on the circumstances.
2. The formal aspects are truly optional and at the discretion of the instructor. The main goal of including this treatment should be to illustrate the *mathematical modeling* component of the course. The suggested syllabus uses CCS – Calculus of Communicating Systems – by Robin Milner. However, any other useful system – e.g., Petri Nets, CSP (Communicating Sequential Processes – C.A.R. Hoare) etc. – can be used.

Contributor/s. Abhijat Vichare (`https://www.linkedin.com/pub/abhijat-vichare/2/822/828`)

3.13 E006 High-Performance Computing

Credits. 5

Prerequisites. C110 (Sec. 2.8) or equivalent and an appropriate programming language, or as defined by the instructor/s

Potentially Dependent Courses. None

Attributions. CBC; C, T, L, S

Rationale, Outlook, Purpose, Objectives, and Goals.

Syllabus.

1. HPC programming platforms. Implicit Parallelism: trends in microprocessor architectures, memory system performance limitations, control structure of HPC platforms, communication model, physical organization, architecture of parallel computer, interconnection networks, network topologies, static, dynamic interconnection networks, cache coherence in multiprocessor systems, communication costs, message passing costs
2. Programming using message-passing. Principles of message passing programming, send and receive operations, blocking and non-blocking message passing, Message passing interface, introduction to MPI routines, data types, concept of communicators, communication domain, communicator groups, creating topologies using MPI, overlapping communication with computation, MPI syntax for frequently used communication calls related to send, receive, barrier, broadcast, reduction, prefix, gather, scatter, all-to-all communication. programs with MPI for addition/multiplication of list of random numbers, matrix-matrix multiplication, bubble sort, shell sort, quick sort, bucket sort, sample sort etc.
3. Parallel algorithm design. Decomposition, tasks, dependency graphs, granularity, concurrency, task interaction, processes and mapping, decomposition techniques: recursive, data, speculative, hybrid, characteristics of tasks and intertask interactions, mapping techniques for load balancing, static mapping, dynamic mapping, methods for containing interaction overheads, parallel algorithm models: data parallel, task-graph, work pool, master-slave, producer-consumer or pipeline model
4. Basic communication operations. Personalized Communication, Collective Communication, Collective communication operation algorithms on ring, mesh, hypercube topologies and their cost analysis, improving speed of communication operations
5. Parallel algorithms for linear algebra. Matrix-vector multiplication with 1-D, 2-D partitioning, matrix-matrix multiplication: simple algorithm (1-D partitioning), Cannon's algorithm (2-D partitioning), DNS algorithm (3-D partitioning), solving system of linear equations with simple Gaussian elimination algorithm (1-D partitioning), 1-D partitioning with pipelined communication and computation, 2-D partitioning with pipelined communication and computation, Gaussian elimination with partial pivoting, solving a triangular system with back substitution, parallel algorithm for Jacobi's iterative method and Gauss-Seidel iterative method for solving system of linear equations.
6. Analytical modeling of parallel programs. Overheads in parallel programs, performance metrics such as execution time, total parallel overhead, speedup, efficiency, cost, effect of granularity on performance, scalability, scaling characteristics, isoefficiency metric of scalability, cost-optimality, isoefficiency function, degree of concurrency and isoefficiency function, minimum execution time, minimum cost-optimal execution time, asymptotic

analysis of parallel programs, other scalability metrics Cost analysis of parallel programs developed in the course work.

Suggested Texts/References.

1. Ananth Grama, Anshul Gupta, George Karypis and Vipin Kumar, *Introduction to Parallel Computing.* Pearson Education, 2004.
2. V. Rajaraman and C. Siva Ram Murthy, *Parallel Computers: Architecture and Programming.* Prentice-Hall India, 2000.
3. Ian Foster, *Designing and Building Parallel Algorithms.* Addison-Wesley, 1995.
4. V. Rajaraman, *Elements of Parallel Computing.* Prentice Hall, 1990.
5. Barry Wilkinson and Michael Allen, *Parallel Programming: Techniques and Applications Using Networked Workstations and Parallel Computers.* Pearson India, .
6. Michael T. Heath, *Scientific Computing.* Tata McGraw-Hill, .
7. Michael Quinn, *Parallel Programming in C with MPI and OpenMP.* Tata McGraw-Hill, .

Notes on Pedagogy.

Contributor/s. Vaishali Shah (https://www.researchgate.net/profile/Vaishali_Shah10)

3.14 E007 Advanced Data Analysis

Credits. 2

Prerequisites. C206 (Sec. 2.13) or equivalent, or as defined by the instructor/s

Potentially Dependent Courses. None

Attributions. CBC; C, T, L, S

Rationale, Outlook, Purpose, Objectives, and Goals. Given the basic background in statistical modeling developed in prior courses, this course intends to expose the student to data analysis in a hands-on and problem-centric manner, so as to develop a feel for the challenges involved. At the end of this course, the student is expected to develop understanding about

1. formulating the problem in the light of the (scientific) question being explored;
2. choosing a statistical method that is most appropriate for the problem;
3. adapting or developing computational tools; and
4. evaluating results of the analysis and the statistical models involved critically.

This course is inspired by similarly-spirited courses such as `http://www.stat.cmu.edu/~cshalizi/uADA/12/` and `http://stat.ethz.ch/education/semesters/ss2010/CompStat`.

Syllabus. An assortment of data analysis problems from any selection of fields at the discretion of the instructor/s. Problems should be chosen to reflect the diversity of (scientific) questions probed as well as that of statistical models and methods, such as (but not limited to): simple and multiple linear regression; model selection; finite mixture models; nonparametric methods for regression, density estimation, and classification (kernel methods, smoothing splines, classification and regression trees, additive models, etc.); resampling, bootstrap, and cross-validation methods.

Suggested Texts/References.

1. Cosma Rohilla Shalizi, *Advanced Data Analysis from an Elementary Point of View*. Unpublished, 2014. Available as `http://www.stat.cmu.edu/~cshalizi/uADA/12/lectures/ADAfaEPoV.pdf`.
2. Peter Bühlmann and Martin Mächler, *Computational Statistics*. Unpublished, 2008. Available as `https://stat.ethz.ch/education/semesters/ss2010/CompStat`.

Notes on Pedagogy. An apt name that conveys the spirit of this course better could have been *Delving into Data Dungeons*. This is intended to be a highly hands-on, interactive course involving an appropriate number of individual and group projects by the students. Apart from the projects, additional case studies may be presented by the instructor or guest speakers so as to develop perspective on the challenges involved in real-life data analysis situations.

Contributor/s. Mihir Arjunwadkar (`http://cms.unipune.ac.in/~mihir`)

3.15 E008 Computing with Java

Credits. 2

Prerequisites. C110 (Sec. 2.8) or equivalent, or as defined by the instructor/s

Potentially Dependent Courses. None

Attributions. CBC; L, S

Rationale, Outlook, Purpose, Objectives, and Goals. Upon successful completion of this course, the student will

1. Understand basics of object-oriented programming (OOP), syntax, semantics of Java.
2. Design an algorithm to solve problems of various kinds in modeling and simulation and implement using Java programming language.
3. Debug the code to spot logical errors, exceptions etc.
4. Write reasonably complex Java code/applets for various problems in modeling and simulation.

Syllabus.

1. Overview of Java Programming. Compiling and Running a Java program. Coding Standards. Introduction to Java programming language. Data types, operators and control structures. Primitive data classes.
2. OOAD. Class, object, inheritance, Class Design and Implementation, class responsibilities, data encapsulation, Polymorphism.
3. Exception Handling. Exception handling in Java.
4. Java Generics. generic programming, ready made classes and generic data structures. Generic Algorithms. Enumerations, autoboxing, and annotations. The Collections Framework
5. Threads. Thread handling in Java.
6. GUI. AWT library. Window, Controls, events, callback, event handlers, frames, graphics, image handling.
7. Applets. Application development and animations.

Suggested Texts/References.

- Herb Schildt, *Java: The Complete Reference*. McGraw Hill Education (India) Private Limited, 2013.
- Various web resources; especially, http://docs.oracle.com/javase/7/docs/api/.

Notes on Pedagogy.

Contributor/s. Bhalchandra Gore (http://cms.unipune.ac.in/~bwgore)

3.16 E009 Theory of Computation

Credits. 5

Prerequisites. As defined by the instructor/s

Potentially Dependent Courses. None

Attributions. CBC; C, T, L, S

Rationale, Outlook, Purpose, Objectives, and Goals. This is an introduction to the formal theory of computation and its modeling applications.

Syllabus.

1. Introduction to Languages. Symbols, strings, words and languages. Symbolic dynamics, dynamics as language. Examples of languages. Finite representation of languages. String induction principles.
2. Finite Automata. Functions as tables: introduction to theory of automata. Regular expressions and languages. Equivalence and simplification of regular expressions. Finite automata and labeled paths. Isomorphism of finite automata. Algorithms for checking acceptance and finding accepting paths. Simplification of finite automata. Proving the correctness of finite automata. Empty-string finite automata. Nondeterministic finite automata. Deterministic finite automata. Closure properties of regular languages. Transfer matrices and finite automata. Solving real-life problems with finite automata.
3. Context-Free Grammars. Examples of languages which are not regular. State minimization. The pumping lemma for regular languages. Context-free grammars, parse trees, stacks and queues. Dynamical systems generating context-free languages. Functions with "internal memory" and push-down automata. Isomorphism of grammars. Derivations, Converting between parse trees and derivations. Simplification and ambiguity of grammars. Determinism and parsing. Pumping lemma for context free grammars. Chomsky normal form. A parsing algorithm.
4. Turing Machines. Examples which are not context free. Chomsky hierarchy. Another look at symbolic dynamics and coding theory for examples of dynamical systems in various levels of Chomsky hierarchy. Computing with dynamical systems. Functions with "external memory" and Turing machines. Computing with Turing machines. Extensions of Turing machines. Random access Turing machine. Non-deterministic Turing machines. Chaotic systems as Turing machines.
5. Universal Turing Machines, Complexity, Computability. Universal Turing machines. Church-Turing thesis. Halting problem. Undecidable problems. Tiling problem and the Potts model. Computability and complexity theory. Classes P and NP. Cook's theorem and P-NP completeness theorems.
6. Formal Computation as a Modeling Paradigm. A judicious selection of topics such as: symbolic dynamics of dynamical systems, biological sequences and stochastic grammars, computational musicology, computational linguistics, natural language processing, L-systems, automata in game theory, etc.

Suggested Texts/References.

1. H. Lewis and C. Papadimitrion, *Elements of Theory of Computation.* Prentice-Hall, 1998.
2. V. E. Krishnamurthy, *Introductory Theory of Computer Science.* Springer-Verlag, 1985.

Notes on Pedagogy. Formal development should be coupled with adequate emphasis on modeling applications, and preferably some hands-on work in any form such class projects, etc.

Contributor/s. Ashutosh Ashutosh (https://www.linkedin.com/profile/view?id=198572337), Abhijat Vichare (https://www.linkedin.com/pub/abhijat-vichare/2/822/828)

3.17 E010 Functional Programming

Credits. 2

Prerequisites. As defined by the instructor/s

Potentially Dependent Courses. None

Attributions. CBC; L, S

Rationale, Outlook, Purpose, Objectives, and Goals. Upon successful completion of this course, the student will be able to

1. Understand basics of functional programming.
2. Design an algorithm to solve problems of various kinds in modeling and simulation and implement using functional language.
3. Debug the code to spot logical errors, exceptions etc.
4. Write reasonably complex code for solving various problems in modeling and simulation.

Syllabus.

1. Introduction to theoretical framework for describing functions and their evaluation, variable binding and substitution, effective computability, Turing machine and Lambda calculus.
2. Dynamically and Statically Typed Functional Languages.
3. Other domain specific programming languages based on/implementing functional programming concepts; e.g, `R`, `SQL`, etc.
4. How does the chosen programming language (LISP/Haskell) implement the above concepts?
5. First-class and higher-order functions, pure functions, recursion.
6. Comparison with other programming paradigms.
7. Declarative *vs.* imperative, using pipelines, Computation by expression evaluation.
8. Reliability.
9. Syntax, coding style and other aspects of programming in the chosen functional language.

Suggested Texts/References. As recommended by the course instructor.

Notes on Pedagogy. Any functional programming language like such as `LISP`, `Scheme`, or `Haskell` can be chosen for this course. The concepts of functional programming, how they are used in the chosen language and programming in that language for M&S problem-solving should be clearly brought out by the instructor.

Contributor/s. Bhalchandra Gore (`http://cms.unipune.ac.in/~bwgore`),
Charulata Patil (`charulata@cms.unipune.ac.in`)

3.18 E011 Computing with `Python`

Credits. 2

Prerequisites. C110 (Sec. 2.8) or equivalent, or as defined by the instructor/s

Potentially Dependent Courses. None

Attributions. CBC; L, S

Rationale, Outlook, Purpose, Objectives, and Goals. `Python` offers a powerful means for M&S via its intuitive object-oriented framework and in-built libraries. The goal of this course is not the introduction to `Python` but demonstration of how `Python` can be used in mathematical modeling. At the end of the course students are expected to develop `Python` programs to implement intermediate level of mathematical models.

Syllabus.

1. Introduction. 'Hello world' program, Understanding `Python` shell and environment. Running the codes.
2. Data and Control Structures. Data types and objects. Introduction to conditionals (`if`) and loops (`for` and `while`). Loading packages.
3. Functions. Defining and using functions. Using built-in functions. Recursion.
4. Advanced Mathematics. Examples of using `numpy, scipy`. Simple 2D and 3D plotting.
5. User Input. Reading and writing the data from/to terminal and files. Formatting output. Converting data types. `try-except` construct.
6. Strings and Lists. Operations on stings. List, list-comprehensions, list of lists.
7. Classes. User defined classes, objects, methods, and functions.

Suggested Texts/References.

1. David Beazley, Brian K. Jones, *Python Cookbook*. O'Reilly Media, 2013.
2. Mark Lutz, *Programming Python*. O'Reilly Media, 2010.
3. The Official Python Documentation (`https://docs.python.org/`).

Notes on Pedagogy. Since the focus of the course is using `Python` for modeling, the syntax could be introduced as a necessity to solve the given problem. Ideally the aim of the course could be to have one `Python` project at the the end of the term, and necessary syntax and other tools are taught as the requirement along the development.

Contributor/s. Bhalchandra Pujari (`http://cms.unipune.ac.in/~bspujari`)

3.19 E013 Computing with R

Credits. 1

Prerequisites. CP (Sec. 1.3)

Potentially Dependent Courses. C206 (Sec. 2.13), C303 (Sec. 2.17), E003-1 (Sec. 3.7), E003-2 (Sec. 3.8), E007 (Sec. 3.14)

Attributions. CBC; L

Rationale, Outlook, Purpose, Objectives, and Goals. The R (`http://cran.r-project.org/`) statistical computing environment, built around the S programming language, is rich in computational statistics primitives. It is open-source and supported by an ever-growing community of users and contributors. It allows a variety of programming styles from quick-and-dirty explorations to elaborate imperative, procedural, object-oriented, and functional coding. It is ideally suited for statistical modeling and data analysis, graphics and visualization, as well as a platform for teaching/learning probability and statistics through hands-on exploration. As such, R is a must for any broad-based M&S curriculum with a statistical modeling/data analysis component. Goals: proficiency in computational problem-solving using R; specifically, decent algorithmic, coding, and scripting skills.

Syllabus.

1. Overview of R and S. History of R. Why use R? When not to use R? GUIs for R. Invoking and exiting the R interpreter environment. Getting help and finding information. `demo()`. The six atomic types. Assignment operators. Standard arithmetic and logical operators. Comments. Conditionals and loops. Parenthesis and braces. Expressions. Every expression has a value. Common composite data types: `vector`, `list`, `matrix`, and `data.frame`. The elementwise operations rule for `vector` and related container types. `functions`. Writing and executing R scripts: `source()` and `Rscript`.
2. Case studies illustrating R capabilities, in-built functions, and common packages. Overview of R graphics. Probability distributions and random number generators. Creating numerical and graphical data summaries, and exploratory data analysis. Complex numbers, numerical methods, etc. Character strings. Set operations. Interface to the operating system shell. Data input and output.
3. Installing R and R packages locally into a linux user account. Installing R from source: `configure` – `make` – `make install` sequence. Installing packages: `install.packages()` and the `R CMD INSTALL` mechanism.
4. Migrating from C to R. Automatic type identification in an assignment vs. explicit declaration of data type. `;` and `\n` as expression terminators. Explicit loops vs. vectorization.
5. Getting performance from R codes. Coding style guidelines. Explicit loops vs. vectorization. The `compiler` package. Debugging and profiling tools. Interfacing with `C`, `C++`, `fortran`.
6. Hands-on explorations using R. Any reasonable set of hands-on problems designed to enhance computational problem-solving and algorithmic abilities. Such problems may be related to M&S in general, or specifically to topics from other courses (e.g., probability theory, statistical inference) in the programme or the instructor's field of expertise.

Suggested Texts/References.

1. W. N. Venables, D. M. Smith, and the R Development Core Team, *An Introduction to R*. The R Project, latest available edition. http://cran.r-project.org/doc/manuals/R-intro.html
2. John Verzani, *Using R for Introductory Statistics*. Chapman & Hall/CRC, 2005.
3. Daniel Navarro, *Learning Statistics with R: A Tutorial for Psychology Students and Other Beginners*. Self-published, 2013. http://learningstatisticswithr.com/
4. Paul Murrell, *R Graphics*. Chapman & Hall/CRC, 2011.
5. Patrick Burns, *The R Inferno*. http://www.lulu.com/, 2012. Available at http://www.burns-stat.com/documents/books/the-r-inferno/.
6. W. N. Venebles and B. D. Ripley, *Modern Applied Statistics with S-Plus*. Springer, 2002.
7. R. G. Dromey, *How to Solve It By Computer*. Prentice-Hall, 1982.

Notes on Pedagogy. This syllabus is based on an outline for a longer course that was refined over several course deliveries by the contributor (see below). Depending on the background and capabilities of the students, this outline may need to be somewhat diluted or intensified – without compromising upon the essentials and goals for the course. Apart from familiarizing a student with R, a major emphasis of this course is on tinkering and exploration, on computational problem-solving, and on translating a problem into a computational framework leading to either a solution or a better understanding of the problem, and on how R can be used as a M&S tool, and for exploring/visualizing probability and statistics concepts. Assignments often consist of problems that are exploratory in nature (e.g., illustrating formal results that may be difficult to grasp, such as the central limit theorem; see C106 (Sec. 2.7)), or require a student to understand an algorithm from its plain-English or pseudocode description (e.g., generating the next permutation given a permutation of n objects). Examinations may consist of problems not necessarily discussed in the class: Here, adequate information about the method of solution or algorithm is provided.

Contributor/s. Mihir Arjunwadkar (http://cms.unipune.ac.in/~mihir)

3.20 E014 Computing with `MATLAB/Scilab`

Credits. 1

Prerequisites. CP (Sec. 1.3)

Potentially Dependent Courses. C204 (Sec. 2.11), C301 (Sec. 2.15), E002-1 (Sec. 3.4), E002-2 (Sec. 3.5)

Attributions. CBC; L

Rationale, Outlook, Purpose, Objectives, and Goals. `MATLAB` and its open-source parallel `Scilab` are popular, powerful, and flexible platforms for numerical and symbolic computation, visualization and graphics, etc., and are rich in computational primitives for diverse fields from digital signal processing to statistics. This course aims at developing an intermediate skill level in writing scripts, performing calculations, using the command line, importing data from files, plotting data, and integrating with other programming languages such as `C`.

Syllabus.

1. Introduction. Environment. Workspaces. General syntax.
2. Numerics. Creating matrices. Matrix operations. Sub-matrices. Statistical operations. Polynomials, differential equations.
3. Plots. Plotting graphs for 2D, 3D functions. Various types of plots.
4. Programming. Functions, `Scilab/MATLAB` programming language, Script files and function files.
5. I/O. Reading, writing data in various formats.
6. Interfacing with programming languages such as `C`.

Suggested Texts/References.

1. Amos Gilat, *MATLAB: An Introduction with Applications*. Wiley, 2008.
2. Mathews and Fink, *Numerical Methods Using MATLAB*. Pearson, 2004.
3. J. C. Polking and D. Arnold, *Ordinary Differential Equations using MATLAB*. Pearson, 2003.
4. An extensive comparison of `MATLAB` and `Scilab`: `http://www.professores.uff.br/controledeprocessos-eq/images/stories/Comparative-Study-of-Matlab-and-Scilab.pdf`

Notes on Pedagogy. Case studies and problems used for introducing `MATLAB/Scilab` should ideally be derived from other courses (e.g., differential equations C104 (Sec. 2.5), C105 (Sec. 2.6)) running concurrently.

Contributor/s. Bhalchandra Pujari (`http://cms.unipune.ac.in/~bspujari`)

3.21 E015 Computing with C

Credits. 2

Prerequisites. CP (Sec. 1.3)

Potentially Dependent Courses. E005 (Sec. 3.12), E006 (Sec. 3.13), E017 (Sec. 3.23)

Attributions. CBC; L

Rationale, Outlook, Purpose, Objectives, and Goals. Upon successful completion of this course, the student is expected to be able to

1. Understand basics of procedural/functional programming, syntax, semantics of C.
2. Design an algorithm to solve problems of various kinds in modeling and simulation and implement using C programming language.
3. Debug the code to spot logical errors, exceptions etc.
4. Write reasonably complex C code for solving various problems in modeling and simulation.

Syllabus.

- ANSI C. Syntax, data types, concept of void, variables, operators, expressions and statements, character input and output, console input and output, inclusion of standard header files, pre-processor directives.
- Control flows. If-else, for, while, do-while, switch-case, break and continue, code blocks and nesting of blocks.
- Functions. Basics of functions, return statement, recursion, function blocks, static variables
- Memory management. Dynamic versus static memory allocation, freeing memory, arrays, memory layout of multidimensional arrays.
- Pointers. Concept of pointers, pointer arithmetic, pointers versus arrays, array of pointers.
- Program compilation and debugging techniques. Introduction to tools like `gdb` together with `ddd`, `GNU make` and profiler `gprof`. Code organization across files. Version/revision control using `svn` or `git`.
- Structures and Unions. Structures and unions, bit fields, typedef, self referential structures, their use in link-list, queue, stack *etc.*
- Input and output in C. Files, file operations.

Suggested Texts/References.

- Kernighan and Ritchie, *The C Programming Language.* PHI, 1990.
- R. L. Kruse, B. P. Leung, C. L. Tondo, *Data Structures And Program Design In C.* Pearson Education, 2007.

Notes on Pedagogy. Although finite-precision arithmetic is covered at length in the course C201 (Sec. 2.10), the student may be exposed here to the bare-basics of finite-precision representations and arithmetic if time permits, and at the discretion of the instructor.

Contributor/s. Bhalchandra Gore (`http://cms.unipune.ac.in/~bwgore`)

3.22 E016 Transforms

Credits. 2

Prerequisites. C101 (Sec. 2.2), C102 (Sec. 2.3), C103 (Sec. 2.4), C201 (Sec. 2.10), a programming language

Potentially Dependent Courses. E001-1 (Sec. 3.2), E001-2 (Sec. 3.3)

Attributions. CBC; C, T, L

Rationale, Outlook, Purpose, Objectives, and Goals. Integral and discrete transforms are often useful in transforming a complex problem into a simpler one. Moreover, insights about the system can be obtained through transforms. Needless to say that integral and discrete transforms are vital tools in the modeling and simulation premises. The goal of this course is to introduce students to a few commonly used transforms with substantial emphasis on Fourier transform. A significant computing aspect is also expected. Students should be able to write codes for some of the transforms. Students are also expected to learn to analyze the results of transformed signals through computational platforms such as `matlab/scilab/R`.

Syllabus.

1. Introduction and background. Brief introduction to vector spaces, function spaces and basis sets. Special functions. Function parity. Concepts in complex analysis. Kernel of an integral transform.
2. Fourier series. Periodic functions. Fourier series in trigonometric as well as complex exponent representation. Functions with arbitrary period. Solving differential equations with Fourier series.
3. Fourier transform. Fourier integrals. Fourier sine/cosine transforms. Fourier transform. Inverse Fourier transform. Properties of Fourier transform. Convolution. Applications. Solving differential equations using Fourier transform, Power spectrum and its interpretation.
4. Discrete Fourier transform and fast Fourier transform. Discretization. Sampling. Nyquist sampling. Discrete Fourier transform. Properties. Matrix representation of Discrete Fourier transform. Fast Fourier transform. Comparison.
5. Laplace transform. Laplace transform. Properties. Inverse transform using partial fractions, convolution and complex integration. Applications. Solution of differential equations.
6. Z-transform. Definition of Z-transform. Properties. Inverse Z-translations using complex integral methods. Difference equations.
7. Wavelet transform (Optional). Limitations of Fourier transform. Introduction to wavelets and family of wavelets. Translations and scaling. Continuous and Discrete wavelet transform. Haar scaling and wavelet functions. Functions spaces. Decomposition using Haar bases. General wavelet system. Daubechies wavelets. Multiresolution analysis. Analysis of output of wavelet transforms.

Suggested Texts/References.

1. L. C. Andrews and B. K. Shivamoggi, *Integral Transforms for Engineers.* Prentice-Hall of India, 2003.

2. Erwin Kreyszig, *Advanced Engineering Mathematics.* Wiley India, 2014.
3. R. N. Bracewell, *The Fourier transform and its applications.* Tata McGraw-Hill, 2003.
4. L. Devnath and D. Bhatta, *Integral transforms and their applications.* Chapman and Hall/CRC, 2010.
5. K. P. Soman and K. I. Ramchandran, *Insights into wavelets - From theory to practice.* Prentice-Hall India, 2005.
6. Online coursework such as Brad Osgood (Stanford), Alan Oppenheim (MIT), etc.

Notes on Pedagogy. An instructor may choose to alter the order to introduce the transforms. However it is logical to start with Fourier series for periodic function, followed by Fourier transform for non-periodic function. The limitation of Fourier transform to resolve the function in both frequency and time domains leads to use of Wavelet and on the other hand Laplace transform can be viewed as 'generalized' Fourier transform, which takes into account the real part of frequency. Similarly Z-transform can be seen as generalized discrete Fourier transform.

A significant emphasize is expected to be on students developing their own codes for various transforms.

Contributor/s. Bhalchandra Pujari (`http://cms.unipune.ac.in/~bspujari`)

3.23 E017 Difference Equations

Credits. 2

Prerequisites. C101 (Sec. 2.2)

Potentially Dependent Courses. E002-1 (Sec. 3.4), E002-2 (Sec. 3.5)

Attributions. CBC; C, T

Rationale, Outlook, Purpose, Objectives, and Goals. As with C104 (Sec. 2.5) and C105 (Sec. 2.6), this course aims at developing a modeling formalisms for describing change.

Syllabus.

1. Definitions of difference equations (ordinary and partial) dependent and independent variables, order and degree, types (linear, quasilinear) elliptic, hyperbolic and parabolic difference equations. Types of side conditions for each.
2. Translation operator and algebraic methods to solve degree one homogeneous and inhomogeneous difference equations. Similarity with ODE methods. Methods using the Z transform.
3. Numerical methods of solution of ordinary difference equations. Examples of usage of data structures and creation of algorithms in this context. Difference equations that result from discretization of differential equations (two examples like Euler stepping and Runge Kutta method).
4. Translation operators for partial difference equations and algebraic methods for solving basic forms of partial difference equations analytically.
5. Numerical methods for partial difference equations, classification as sweeping and stepping methods. Examples of usage of data structures and creation of algorithms in this context. Difference equations that result from discretization of partial difference equations (examples of finite difference method only).
6. Application of difference equations in population dynamics and finance.
7. Introduction to theoretical concepts of stability, oscillation and asymptotic behaviour of difference equations and their solutions.

Suggested Texts/References.

1. Saber N. Elaydi, *Introduction to Difference Equations*. Springer, 1999.
2. Ronald E. Mickens, *Difference Equations*. CRC Press, 1991.
3. Walter C. Kelley and Allan C. Peterson, *Difference Equations: An Introduction with Applications*. Academic Press, 2001.
4. *Mathematical Modelling Through Difference Equations*, Chapter 5, in: J. N. Kapur, *Mathematical Modelling*. New Age International Publishers, 2008.
5. Sui Cun Cheng, *Partial Difference Equations*. Taylor and Francis, 2003.

Notes on Pedagogy.

Contributor/s. Sukratu Barve (`http://cms.unipune.ac.in/~sukratu`)

3.24 E018-1 Complex Networks 1

Credits. 5

Prerequisites. C103 (Sec. 2.4), C106 (Sec. 2.7), C110 (Sec. 2.8)

Potentially Dependent Courses. E018-2 (Sec. 3.25)

Attributions. CBC; C, T, L

Rationale, Outlook, Purpose, Objectives, and Goals. A large number of real-world systems like Facebook, air-transport, metabolic reactions inside a living cell, and the Internet can be modelled as networks. Also several phenomena like traffic jams, rumour spreading and genetic regulations can be modelled as processes on networks making it an indispensable tool to study 'complex systems'. The theory of complex networks builds on methods borrowed from computer science, physics, statistics, social sciences and many others, and so is highly interdisciplinary. The main goal of this first part is to introduce students with the scope and methods of the field, basic mathematics and algorithms of networks, and a few models of networks. The main book that would be used for the course is *Networks: An introduction* by Mark Newman (2nd edition). Students are expected to implement the algorithms in a language of their choice instead of using packages available for the analysis of networks.

Syllabus.

1. Introduction to networks. Empirical networks: Technological, Social, Information, Biological.
2. Mathematics of networks.
   (a) Mathematical representation, Adjacency matrix.
   (b) Weighted, directed, bipartite and temporal/dynamic networks.
   (c) Trees, planar graphs.
   (d) Degree, paths, components.
3. Network quantification using local values.
   (a) Degree distribution, local clustering distribution.
   (b) Centrality values and distributions: degree, eigenvector, katz, pagerank, closeness, betweenness.
   (c) Transitivity, reciprocity, similarity.
4. Large-scale structure.
   (a) Global clustering coefficient.
   (b) Components in undirected and directed networks.
   (c) Scale-free networks, detection of power-laws.
   (d) Assortative mixing and modularity.
5. Network algorithms.
   (a) Basics of time complexity of algorithms.
   (b) Storing network data: adjacency matrix, adjacency list, edge list.
   (c) Algorithms for degree distributions, clustering coefficients.
   (d) Eigenvector centrality calculation using iterative updating.
   (e) Breadth-first search, Dijkstra's algorithm.

6. Overview of simple models of network formation.

   (a) Erdos-Renyi graph (Emphasis on small path lengths).
   (b) Watts-Strogatz model (Emphasis on small-world effect).
   (c) Price/Barabasi-Albert graph (Emphasis on emergence of power-law through preferential attachment), Master equation.
   (d) Computer simulations of the three models.
   (e) (Optional) Further properties of the preferential attachment models. Extensions of preferential attachment models. Vertex copying models.

Suggested Texts/References.

1. Mark Newman, *Networks: An introduction.* Oxford University Press (New York), 2018 or latest.
2. Harry Crane, *Probabilistic Foundations of Statistical Network Analysis.* Chapman and Hall/CRC, 2018.
3. Albert-László Barabasi, *Network Science.* Cambridge University Press, 2016.
4. Guido Caldarelli, *Scale-Free Networks: Complex Webs in Nature and Technology.* Oxford University Press UK, 2013.
5. S.N. Dorogovtsev and J.F.F. Mendes, *Evolution of Networks: From Biological Nets to the Internet and WWW.* Oxford University Press (Oxford), 2013.

Notes on Pedagogy. Knowledge of at least one programming language (`Python`, `R`, `C/C++`, etc.) or computational environment (`Matlab/SciLab`, `Mathematica`, etc.) is required.

Contributor/s. Snehal Shekatkar (`https://inferred.co/`)

3.25 E018-2 Complex Networks 2

Credits. 5

Prerequisites. E018-1 (Sec. 3.24), C201 (Sec. 2.10)

Potentially Dependent Courses. E018-2 (Sec. 3.25)

Attributions. CBC; C, T, L

Rationale, Outlook, Purpose, Objectives, and Goals. The part 2 of the course deals with several intermediate to advanced topics in the theory of complex networks. Mastering the material presented here will prepare the students to handle several real-world problems involving networks. In this part, a student need to have some familiarity with linear algebra and complex analysis. Some familiarity with statistical physics would be advantageous but is not assumed.

Syllabus.

1. Simple processes on networks.
   - Diffusion on networks and graph Laplacian.
   - Properties of graph Laplacian and algebraic connectivity.
   - Random walks on networks.
2. Graph partitioning and community detection.
   - Partitioning vs community detection.
   - Kernighan-Lin algorithm.
   - Spectral partitioning.
   - Kernighan-Lin-Newman algorithm for modularity maximization.
   - Spectral modularity maximization.
   - Newman-Girvan algorithm.
   - (Optional) Problems with modularity maximization: NP-hard, degeneracy, noise, resolution limit. Discussion about the statistical inference on networks.
3. Random graphs.
   - The concept of random graphs.
   - Erdos-Renyi (ER) or $G(n, p)$ graph: mean edges, clustering (emphasis on the asymptotically 0 value), degree-distribution.
   - Concept of the giant component (GC) in a random graph, GC in ER graph.
   - path lengths in $G(n, p)$.
   - (Optional) Small components in $G(n, p)$, cavity method. Complete distribution of small components.
4. Random graphs with arbitrary degree-distributions.
   - Generating functions and their powers.
   - The configuration model.
   - Excess degree distribution, friendship paradox.
   - (Optional) Random graphs with a given expected degree. Generating functions for the small components in the configuration model. The giant component in the configuration model.

5. Other network models.
    - The stochastic block model and its variants.
    - (Optional) The small-world model. Exponential random graphs.
6. Percolation and network resilience.
    - Percolation (edge and vertex).
    - Computer algorithms for percolation.
    - simulations of uniform deletion of vertices in ER and BA networks.
    - (Optional) uniform removal of vertices in the configuration model. Non-uniform removal of vertices.

Suggested Texts/References.

1. Mark Newman, *Networks: An introduction.* Oxford University Press (New York), 2018 or latest.
2. Albert-László Barabasi, *Network Science.* Cambridge University Press, 2016.
3. Guido Caldarelli, *Scale-Free Networks: Complex Webs in Nature and Technology.* Oxford University Press UK, 2013.
4. S.N. Dorogovtsev and J.F.F. Mendes, *Evolution of Networks: From Biological Nets to the Internet and WWW.* Oxford University Press (Oxford), 2013.
5. Mark Newman, Albert-Lászlό Barabasi and Duncan Watts, *The structure and Dynamics of Networks.* New Age International Pvt Ltd, 2010.

Notes on Pedagogy. Knowledge of at least one programming language (`Python`, `R`, `C/C++`, etc.) or computational environment (`Matlab/SciLab`, `Mathematica`, etc.) is required.

Contributor/s. Snehal Shekatkar (`https://inferred.co/`)

3.26 E019-1 Astrostatistics 1

Credits. 5

Prerequisites. C206 (Sec. 2.13)

Potentially Dependent Courses. E019-2 (Sec. 3.27)

Attributions. CBC; C, T, L

Rationale, Outlook, Purpose, Objectives, and Goals. Outlook and rationale for this course can be summarized as follows:

> With data rates, sizes, and complexities soaring up high over the coming decades, meaningful investigation into a scientific question will require fresh ways of identifying patterns and structure in the data using sophisticated statistical and computational methodologies. (Indeed, the 21st century science has been aptly described as large datasets, complex questions science (Efron, 2011).) Technology development, which is essential for progress of science, also necessitates methodological development for its efficient and effective use. It is important to remember that it is the nature of the data and scientific questions being addressed which should dictate the method, and not vice versa (Arjunwadkar, Kashikar, and Bagchi, J. Astrophys. Astr. (2016) 37:28).

This course aims at introducing students with adequate background in statistics and computation – but not necessarily in astronomy and astrophysics (A&A) – to the field of astrostatistics. See the Notes on Pedagogy section of this syllabus document for more details.

Syllabus.

1. A crisp introduction to astronomy and astrophysics (outreach+ / undergraduate− level). Overview/survey of astronomical phenomena and their physics at a very basic level, essential terminology, and historical perspective as appropriate.
    (a) Basics of Astronomy: Observing the sky (naked eye, telescopes, and other instruments), measuring distances, distance scales, units, etc.
    (b) Basics of Astrophysics: blackbody radiation, stars, galaxies and their distribution, scaling relations.
    (c) Astronomical data: Nature of data-gathering and measurement processes together with discussion of the underlying statistical assumptions. Sources of errors in measurement and calibration.
2. Density estimation.
    (a) Histograms, bias-variance tradeoff, optimal bin width, confidence bands.
    (b) Kernel Density Estimation: Univariate and multivariate, optimal bandwidth via cross-validation, confidence bands.
    (c) Adaptive smoothing: Adaptive kernel estimators, nearest-neighbour estimators.
    (d) Density estimation via normal mixtures: estimation, model selection, and bootstrap confidence intervals on parameters.
    (e) Nonparametric density estimation via orthogonal functions, and confidence sets.

3. Regression and model selection.

   (a) Generalities: Regression and regression function, quadratic prediction risk and $r(x) = \mathsf{E}(Y|X = x)$ as its minimizer, bias-variance decompositions.
   (b) Simple and multiple linear regression, least squares: unweighted and weighted, maximum likelihood, confidence intervals and bands, prediction bands, hat matrix, simple tests for coefficients, diagnostics, collinearity, transformations as a way to conform to assumptions better, linearity as linearity in the coefficients.
   (c) Variable/model selection: Training risk, Mallow's C_p, leave-one-out cross-validation, AIC, BIC, MDL, variable selection vs. hypothesis testing. An overview of regularization (ridge regression, LASSO and sparsity). Identifiability.
   (d) Nonparametric regression: Regressogram, running mean / local average smoother, linear smoothers, smoothing (hat) matrix. Choosing the smoothing parameter: Bias-variance tradeoff, predictive risk and its estimators: training risk, leave-one-out-cross-validation risk, generalized cross-validation risk, etc. Kernel regression. Nonparametric regression via orthogonal functions, and confidence sets. Local polynomials. Regularization and splines.
   (e) Robust regression. Nonlinear regression. Handling (measurement) error in the co-variates.

4. Clustering and classification. A crisp overview of supervised, unsupervised, and reinforcement learning, and a subset of topics in the following broad area: Generative and discriminative classifiers. Naive Bayes, Gaussian mixtures, generative adversarial classifiers, K-NN, logistic regression neural networks, decision tree and random forest classifiers. Deep neural networks. K-means, hierarchical and Gaussian mixtures for clustering. Time series classification and clustering. Hypothesis testing for classification. Attribute selection for classification.

Suggested Texts/References.

1. Fiegelson & Babu, *Modern Statistical Methods for Astronomy with R Applications.* Cambridge, 2012.
2. Wall & Jenkins, *Practical Statistics for Astronomers.* Cambridge, 2003.
3. AK Chattopadhyaya, *Statistical Methods for Astronomical Data Analysis.* Springer, 2014.
4. Starck & Murtagh, *Astronomical Image and Data Analysis.* Springer, 2006.
5. Mike Inglis, *Astrophysics is Easy! An Introduction for the Amateur Astronomer.* Springer, 2007.
6. Hastie, Tibshirani, & Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction.* Springer, 2009.
7. Efron & Hastie, *Computer Age Statistical Inference: Algorithms, Evidence, and Data Science.* Cambridge, 2016.

Notes on Pedagogy.

1. This course assumes that the audience have sufficient background in the statistics and computing, but none in A&A.
2. This course in its current form is organized around topics in statistics. Apt and appropriate applications and examples are to be chosen by the instructors from A&A.

3. Depending on the instructors' fields of expertise and the maturity level of the audience, this and/or a follow-up advanced course may also be run entirely in the problem-centric mode which weaves statistics around concrete astronomical problems and their associated datasets. In either mode, instantiations of this course should have a substantial hands-on component (textbook as well as non-textbook) analyzing actual astronomical data sets – from exploratory data analysis all the way upto trying to answer the science questions for which the data was collected.
4. Choice of programming language, computing environment, or software is left to the discretion of the instructors.
5. The above syllabus should be taken as indicative, and not as set in stone. Topics, subtopics, and finer content may be tweaked by the instructors to suite the audience's prior training and know-how. Except for #1 and #2 in the syllabus above, other topics may be presented/discussed in the class in any order. Given the highly multidisciplinary nature of the course, it may be tweaked to match the instructors' areas of expertise in statistics, A&A, astrostatistics, etc., and new developments in these fields.
6. Given the multidisciplinary nature of the course, ideally it should be interspersed with colloquia by astronomers, astrophysicsts, statisticians, and data scientists. Such colloquia may be focused on case studies, specific applications, broad overviews – In short, anything that would help broaden the students' outlook.

Contributor/s.

1. Mihir Arjunwadkar (`http://cms.unipune.ac.in/~mihir`)
2. Akanksha Kashikar (`https://www.researchgate.net/profile/Akanksha_Kashikar`)
3. TV Ramanathan (`https://www.researchgate.net/profile/T_Ramanathan`)
4. VK Jayaraman (`https://scholar.google.co.in/citations?user=GRv1gLQAAAAJ`)
5. Kaustubh Vaghmare (`https://www.researchgate.net/profile/Kaustubh_Vaghmare`)
6. Dhruba J Saikia (`http://mutha.ncra.tifr.res.in/ncra/people/academic/ncra-faculty/djs`)
7. Somak Raychaudhuri (`https://www.researchgate.net/profile/Somak_Raychaudhury`)
8. Dipanjan Mitra (`http://www.ncra.tifr.res.in/ncra/people/academic/ncra-faculty/dmitra`)

3.27 E019-2 Astrostatistics 2

Credits. 5

Prerequisites. E019-1 (Sec. 3.26)

Potentially Dependent Courses.

Attributions. CBC; C, T, L

Rationale, Outlook, Purpose, Objectives, and Goals. Same as that of the prequel course E019-1 (Sec. 3.26). This sequel introduces a few more areas of statistics which are heavily used in statistical data analysis in Astronomy & Astrophysics.

Syllabus.

1. Outliers. A crisp overview of considerations, core concepts, methods, and practice, broadly along the lines of chapters 1 and 2 of #1 in Suggested Texts/References.
2. Missing data. A crisp overview of considerations, core concepts, methods, and practice, broadly along the lines of chapters 1 and 2 of #2 in Suggested Texts/References.
3. Time-series analysis. A crisp overview of considerations, core concepts, methods, and practice, broadly along the lines of chapters 1–6 of #3 in Suggested Texts/References.
4. Spatial statistics. A crisp overview of considerations, core concepts, methods, and practice, broadly along the lines of chapters 1–3 of #6 in Suggested Texts/References.
5. Bayesian inference. Review of axioms of probability theory and Bayes Theorem for conditional probabilities. The Bayesian inferential philosophy: Classical/frequentist vs. subjective probabilities. Bayes theorem and its inferential interpretation as flow of information from prior to posterior via likelihood. Standard examples, such as Bernoulli likelihood with Beta prior, normal likelihood with normal prior; etc. Conjugate priors. Prior distributions: Proper, improper, flat, noninformative. Role of priors in making ill-defined estimation/inference problems well-behaved: E.g., Non-identifiable Gaussian mixture models with too many components (and when data is sparse), etc. Relative importance of prior and likelihood with respect to data size. Posterior distribution as the prime inferential object. Exploring the posterior through simulation. Credible intervals. Bayesian testing. Bayesian model selection. Bayesian vs. classical: Strengths and weaknesses.

Suggested Texts/References.

1. Charu C. Aggarwal, *Outlier Analysis.* Springer, 2013.
2. Little & Rubin, *Statistical Analysis with Missing Data.* Wiley, 2002.
3. Brockwell & Davis, *Introduction to Time Series and Forecasting.* Springer, 2002.
4. Gelman et al., *Bayesian Data Analysis.* CRC Press, 2013.
5. Joseph B. Kadane, *Principles of Uncertainty.* CRC Press, 2011.
6. Cressie, *Statistics for Spatial Data.* Wiley, 1993.
7. Bivand, Pebesma, & Gómez-Rubio, *Applied Spatial Data Analysis With R.* Springer, 2008.

Notes on Pedagogy. See this section for the the prequel course E019-1 (Sec. 3.26).

Contributor/s.

1. Mihir Arjunwadkar (http://cms.unipune.ac.in/~mihir)
2. Akanksha Kashikar (https://www.researchgate.net/profile/Akanksha_Kashikar)
3. TV Ramanathan (https://www.researchgate.net/profile/T_Ramanathan)

3.28 E020 Data Visualization

Credits. 1

Prerequisites. As defined by the instructor

Potentially Dependent Courses. None

Attributions. CBC; C, L, W

Rationale, Outlook, Purpose, Objectives, and Goals. The purpose of this course is to introduce the audience to the field of visual design so as to create awareness and literacy about visual design. The intended audience of this course is assumed to be a lay audience as far as this area is concerned. The expected outcome is awareness and improved understanding of visual display of quantitative information. The design awareness created through introductory lectures should be reinforced through hands-on visualization work using available tools, packages, APIs, libraries, etc.

Syllabus.

1. Introduction to visual design. A guided tour through the world of visual/graphic design so as to create a sense of visual design awareness and literacy. A crisp overview of Edward Tufte's framework for envisioning information.
2. Data visualization. Starting from the all-familiar scatterplots, pair plots, boxplots, histograms, wiremesh plots, contour plots, etc., introduce principles and strategies for visualizing complex / hierarchical / high-dimensional data. A pedagogic approach to data visualization (such as that in Tamara Munzner's book mentioned below) is recommended for this part.
3. Hands-on work. Individual or group miniprojects. Students to survey and choose appropriate tools, libraries, APIs, etc., best-suited for their chosen topic, under the guidance of the instructor/s.

Suggested Texts/References.

1. Robin Williams, *The Non-Designer's Design Book.* Peachpit Press, 2014.
2. Tamara Munzner, *Visualization Analysis and Design.* CRC Press, 2014.
3. Edward Tufte, *Envisioning Information.* Graphics Press, 1990. Edward Tufte, *The Visual Display of Quantitative Information.* Graphics Press, 2001.
4. David McCandless, *Information is Beautiful.* Graphics Press, 2009.
5. Nathan Yau, *Visualize This: The FlowingData Guide to Design, Visualization, and Statistics.* Wiley, 2011.
6. Manuel Lima, *Visual Complexity: Mapping Patterns of Information.* Princeton Architectural Press, 2013.
7. Stephen Few, *Now You See It: Simple Visualization Techniques for Quantitative Analysis.* Analytics Press, 2009.
8. Chun-houh Chen, Wolfgang Härdle, and Antony Unwin, *Handbook of Data Visualization.* Springer, 2008.
9. Phil Simon, *The Visual Organization: Data Visualization, Big Data, and the Quest for Better Decisions.* Wiley, 2014.

Notes on Pedagogy. While this course does not assume any design literacy on part of the audience, the audience may have some prior exposure to design principles from fields such as engineering design, software design, etc. Any appropriate set of books, internet resources, and software such as `ggplot2`, `D3`, etc., may be used as per the instructor's preferences.

Contributor/s.

1. Jitendra Pawagi (https://landconcern.blogspot.com/)
2. Jayant Gadgil (http://sciencepark.unipune.ac.in/about-us.html)
3. Girish Dalvi (http://www.idc.iitb.ac.in/girish/)
4. Bhalchandra Pujari (http://cms.unipune.ac.in/~bspujari)
5. Mihir Arjunwadkar (http://cms.unipune.ac.in/~mihir)