Tutorial
on
Symbolic Computing
with *Mathematica*

Youngjoo Chung

School of Info. and Comm., GIST
ychung@gist.ac.kr

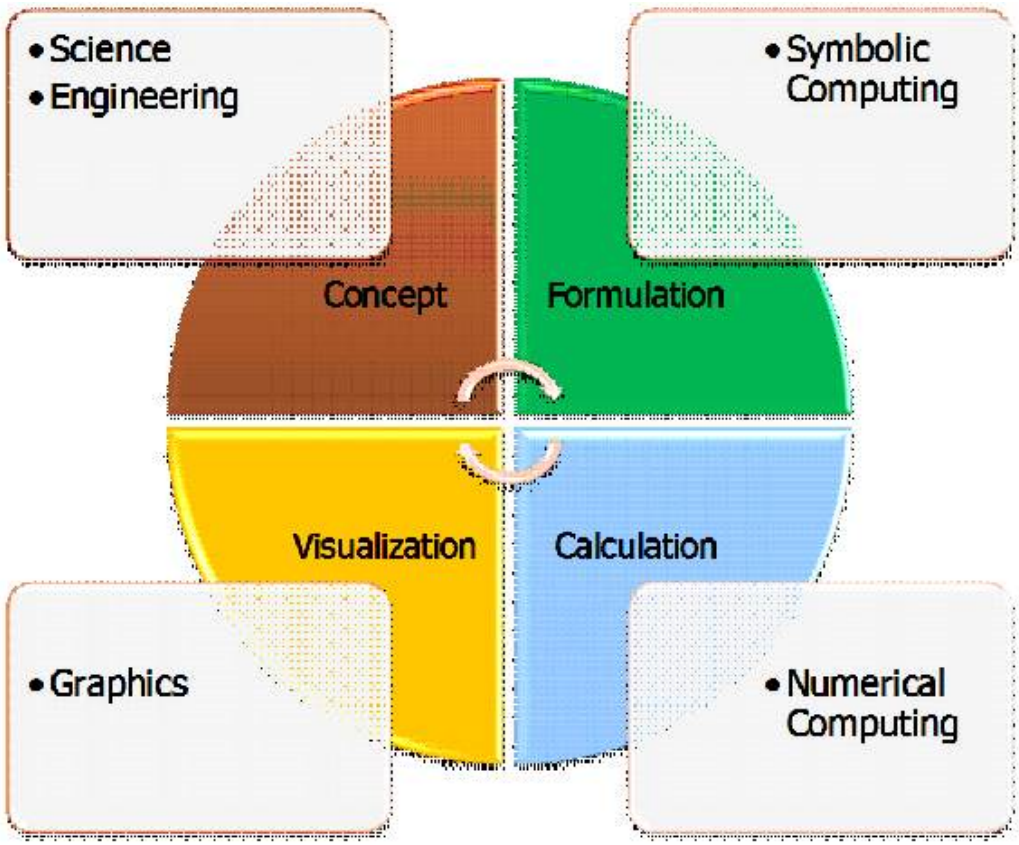
2011. 7. 7

```
<< InitPackages`
```

Symbolic Computation

• Symbolic computation (algebraic manipulation)

– Mathematical computation with variables and constants according to the rules of algebra



• Commercial software

Mathematica
Maple
MathCad
Matlab
Reduce
Macsyma
Scratchpad
Derive

Basic Concepts of *Mathematica*

• *Mathematica* is the world's only fully integrated environment for formulation, calculation and visualization for technical computing.

• The concept of *Mathematica* is to create once and for all a single system that could handle all the various aspects of technical computing in a coherent and unified way.

• The concept of *Mathematica* is to create once and for all a single system that could handle all the various aspects of technical computing in a coherent and unified way.

• Manipulation of the very wide range of objects involved in technical computing using only a fairly small number of basic primitives (over 3,000 built in the kernel of version 7).

• The functionality can be easily extended through user-defined functions and external programs.

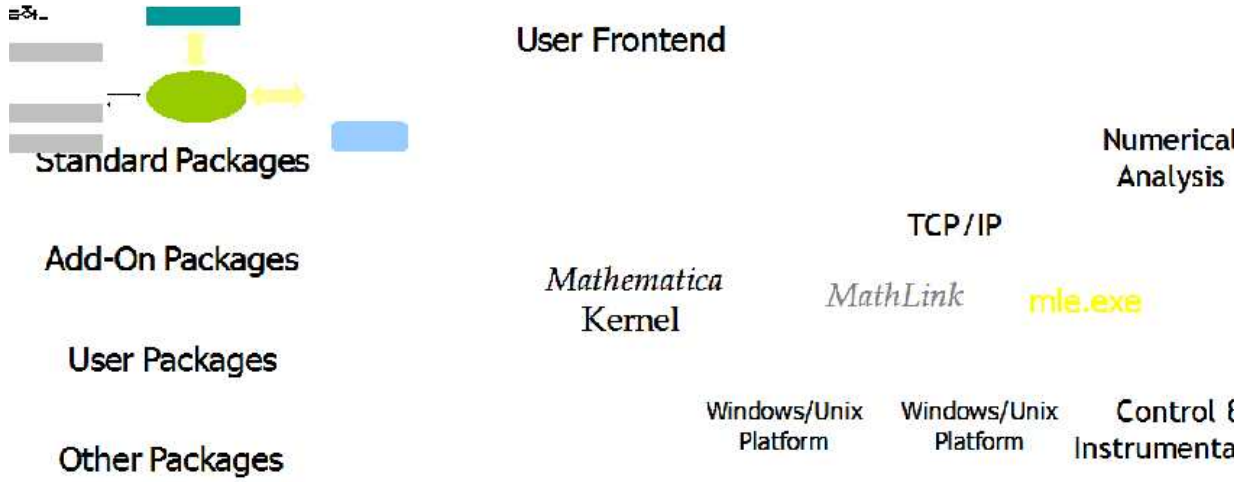
• Platform-independent interactive documents known as notebooks (.nb files)

• System-provided macros and user-defined macros in packages (.m files)

• Interactive and non-interactive sessions

• Communication with external programs using *MathLink*

• *Mathematica* computing environment



– External applications are written in C/C++.
– *Mathematica* is used for both data input and output.
– External applications and *Mathematica* communicate via *MathLink.*
– The functionality can be extended through user-defined functions and external programs.

• More information available at http://www.wolfram.com

Algebraic Calculation

- **Algebraic Calculations (*Mathematica* Tutorial)**
- **Symbolic Calculations (*Mathematica* Tutorial)**
- `Expand`
- `Factor`
- `Apart`
- `Together`
- `Simplify`

Basic Algebra

- **Polynomial Algebra (*Mathematica* Guide)**
- **Polynomial Systems (*Mathematica* Guide)**

Elementary Functions

- **Elementary Functions (*Mathematica* Guide)**

Trigonometric Functions

- **Trigonometric Functions (*Mathematica* Guide)**
- **Trigonometric Expressions (*Mathematica* Tutorial)**

Complex Variables

- **Complex Numbers (*Mathematica* Guide)**
- **Functions of Complex Variables (*Mathematica* Guide)**

Flow Control

- **Conditionals (*Mathematica* Guide)**
- **Loops and Control Structures (*Mathematica* Tutorial)**
- **Flow Control (*Mathematica* Guide)**

Formula Manipulation

Formula Manipulation

- **Formula Manipulation (*Mathematica* Guide)**
- **Manipulating Equations (*Mathematica* Guide)**

User Package Functions

• Mathematica provides a solid foundation for symbolic computing and a number of supporting functions.
– Built-in kernel functions
– User-defined functions to complement the built-in kernel functions

• A mechanism is needed to facilitate the algebraic manipulation of mathematical expressions with the following properties:
– Seamless integration with the computing environment of Mathematica
– Selective targeting of the object to apply functions
– Improved handling of subscripts (a_1), tildes (ã), hats (â), etc.
– On-line setting and clearing of attributes
– Addition of comments
– Improved handling of derivatives, integrals and summations
– Minimal use of variables

• Algebraic manipulation of formulas using symbolic computing
– Mathematica is not a word processor or an equation editor.
– Application of functions
– Substitution using mathematical identities
– Allows focusing on the principles instead of time-consuming and error-prone calculations.
– Good readability
– Minimization of human errors during calculations
– Expressions that closely resemble the traditional mathematical style, e.g., subscripts and vector notations

• A large collection of user-defined functions
– Basic Algebra
– Differentiation
– Integration
– Summation
– Trigonometric Functions
– Complex Variables
– Vectors and Matrices
– Polynomials and Series
– Functional Analysis
– Equations
– Operator Analysis
– Plotting
– Etc.

• A large collection of user-defined functions
  – Basic Algebra
  – Differentiation
  – Integration
  – Summation
  – Trigonometric Functions
  – Complex Variables
  – Vectors and Matrices
  – Polynomials and Series
  – Functional Analysis
  – Equations
  – Operator Analysis
  – Plotting
  – Etc.

• **MPMAF**
  – An abbreviation of **MPMapApplyFunc**
  – The platform for algebraic manipulation of all or parts of an expression

• User-defined functions
  – The function names start with the prefix "MP".
  – Algebraic manipulation of expressions
  – Mathematical identities
  – Can be used separately independent of the macro **MPMAF**, in which case the features provided by the options of **MPMAF** cannot be used.

Basic Algebra

• Expansion

```
Expand[(x + a)^2]
```

$$a^2 + 2\,a\,x + x^2$$

The power exponent must be an integer. Otherwise, use **MPExpandBinomial**.

```
MPExpandBinomial[(x + a)^n, k]
MPEvalSum[%]
```

$$\sum_{k=0}^{n} \frac{a^k\, x^{-k+n}\, n!}{k!\,(-k+n)!}$$

$$(a + x)^n$$

```
MPExpandBinomial[(a + b + c)^n, k, l]
MPEvalSum[%]
PowerExpand[%]
```

$$\sum_{k=0}^{n}\sum_{l=0}^{-k+n}\frac{a^k\,b^l\,c^{-k-l+n}\,n!}{k!\,l!\,(-k-l+n)!}$$

$$c^n\left(\frac{b+c}{c}\right)^n\left(\frac{a+b+c}{b+c}\right)^n$$

$$(a+b+c)^n$$

• Factoring

```
Factor[x^2 + (a + b) x + a b]
```

$$(a+x)\,(b+x)$$

MPFactor

```
MPFactor[a + b, a]
```

$$a\left(1+\frac{b}{a}\right)$$

• Reduction of fractions

```
x^2/(x + a) → ((x + a)^2 - a^2 - 2 a (x + a) + 2 a^2)/(x + a)
```

$$\frac{x^2}{a+x}\to\frac{-a^2-2\,a\,x+(a+x)^2}{a+x}$$

```
Apart[x^2/(x + a), x]
```

$$-a+x+\frac{a^2}{a+x}$$

MPApart

```
MPApart[s_x^2 ϵ_x/(n^2 ϵ_0 - ϵ_x) + s_y^2 ϵ_y/(n^2 ϵ_0 - ϵ_y) + s_z^2 ϵ_z/(n^2 ϵ_0 - ϵ_z), {ϵ_x, ϵ_y, ϵ_z}]
% /. -s_x^2 - s_y^2 - s_z^2 → -1
```

$$-s_x^2-s_y^2-s_z^2-\frac{n^2\,s_x^2\,\epsilon_0}{-n^2\,\epsilon_0+\epsilon_x}-\frac{n^2\,s_y^2\,\epsilon_0}{-n^2\,\epsilon_0+\epsilon_y}-\frac{n^2\,s_z^2\,\epsilon_0}{-n^2\,\epsilon_0+\epsilon_z}$$

$$-1-\frac{n^2\,s_x^2\,\epsilon_0}{-n^2\,\epsilon_0+\epsilon_x}-\frac{n^2\,s_y^2\,\epsilon_0}{-n^2\,\epsilon_0+\epsilon_y}-\frac{n^2\,s_z^2\,\epsilon_0}{-n^2\,\epsilon_0+\epsilon_z}$$

• Separation of variables for solving differential equations

```
dy/dx == 2 e^x y
MPMAF[%, MPSepVars, {All, x, Side → Right},
 MPDiffToInt, {All, x, y, Constant → 1/2 Log[c1]},
 MPEvalInt, All,
 MPSolve, {All, y}]
```

$$\frac{dy}{dx} == 2\, e^{x}\, y$$

$$\frac{\text{Log}[y]}{2} == e^{x} + \frac{\text{Log}[c_1]}{2}$$

$$y == e^{2\, e^{x}}\, c_1$$

Direct solution using **`MPDSolve`**

```
dy/dx == 2 e^x y
MPMAF[%, MPDSolve, {All, y, x, RCN → c1}]
```

$$\frac{dy}{dx} == 2\, e^{x}\, y$$

$$y == e^{2\, e^{x}}\, c_1$$

Trigonometric Functions

• Adding sin and cos functions

```
MPTrigAddSinCos[a Cos[x] + b Sin[x], Sin]
```

$$\sqrt{a^2 + b^2}\ \text{Sin}[x + \text{ArcTan}[b, a]]$$

• Conversion to other functions

```
MPTrigConvert[Tan[x], Cos]
```

$$\sqrt{1 - \text{Cos}[x]^2}\ \text{Sec}[x]$$

• Conversion to half angle

```
MPTrigToHalf[Cos[2 x], Sin]
```

$$1 - 2\, \text{Sin}[x]^2$$

• Conversion to double angle

```
MPTrigToDouble[Cos[x]^2, Cos]
```

$$\frac{1}{2}(1+\text{Cos}[2\,x])$$

Complex Variables

• Asterisk (*) is interpreted as the complex conjugate

```
MPComplexExpand[(ℯ^(u+v))*, u, TargetFunctions → Conjugate]
```

$$\text{e}^{v+u^*}$$

The built-in function `ComplexExpand` gives a rather different (even though equivalent) result.

```
ComplexExpand[Conjugate[ℯ^(u+v)], u, TargetFunctions → Conjugate]
```

$$\text{e}^{v+\frac{1}{2}(u+\text{Conjugate}[u])}\,\text{Cosh}\left[\frac{1}{2}(u-\text{Conjugate}[u])\right]-\text{e}^{v+\frac{1}{2}(u+\text{Conjugate}[u])}\,\text{Sinh}\left[\frac{1}{2}(u-\text{Conjugate}[u])\right]$$

• The exponential form of a complex number

```
MPComplexToExp[x + i y]
```

$$\text{e}^{i\,\text{ArcTan}\left[\frac{y}{x}\right]}\sqrt{x^2+y^2}$$

Polynomials and Series

• Eliminate the second highest order term

```
MPMergePoly[x^3 + a x^2 + c, x]
```

$$-\frac{a^3}{27}+c-\frac{a^2\,x}{3}+\left(\frac{a}{3}+x\right)^3$$

• Transform a polynomial by making a replacement of the variable

```
MPTransPoly[x^3 + a x^2 + c, x, a + x]
```

$$c+a^2\,(a+x)-2\,a\,(a+x)^2+(a+x)^3$$

A case of two variables

```
MPTransPoly[x^2 (y + a), {x, y}, {x - 1, y + 1}]
```

$$a+(-2+2\,a)\,(-1+x)+(-1+a)\,(-1+x)^2+y+2\,(-1+x)\,(1+y)+(-1+x)^2\,(1+y)$$

• Taylor series

```
MPTaylor[Sqrt[1 + x], x, 3]
```

$$1 + \frac{x}{2} - \frac{x^2}{8} + \frac{x^3}{16}$$

```
MPTaylor[Sqrt[1 + (x + Δx)^2], Δx, 1]
```

$$\sqrt{1 + x^2} + \frac{x\,\Delta x}{\sqrt{1 + x^2}}$$

```
MPAFE[y[x], MPTaylor, {All, x - x0, 2, Variables → x}, CS → x0, Head → TildeTilde]
```

$$y[x] \approx y[x_0] + (x - x_0)\, y'[x_0] + \frac{1}{2}(x - x_0)^2\, y''[x_0]$$

Equations

• Adding equations

```
MPAddEqs[{a == b, c == d, e == f}]
```

$$a + c + e == b + d + f$$

• Subtracting equations

```
MPSubtEqs[{a == b, c == d, e == f}]
```

$$a - c - e == b - d - f$$

• Move a subpart of an equation to left- or right-hand side

```
MPMoveEq[x + y == z, y, Right]
```

$$-x + z == y$$

• Eliminating variables from a set of equations

```
MPEli[{x + y == 1, x - y + 2 z == 1, y - 3 z == 3}, {y, z}] // Simplify
```

$$2x == 5$$

• Solve equations and write

```
MPSolve[{x + y == 1, x - y + 2 z == 1, y - 3 z == 3}, {x, y, z}]
```

$$\left\{x == \frac{5}{2},\ y == -\frac{3}{2},\ z == -\frac{3}{2}\right\}$$

Differentiation

• The built-in derivative operator `D`, or ∂, evaluates the expression immediately.

```
D[f[x], x]
```

```
f'[x]
```

The total derivative operator `Dt` assumes all symbols are dependent variables.

```
Dt[a f[x], x]
```

```
Dt[a, x] f[x] + a f'[x]
```

• The partial differential operator $\frac{\partial}{\partial x}$ delays the evaluation until the command **MPEvalD** or **MPExpandD** is encountered.

```
y == ∂^3/(∂x^2 ∂y) (a f + b g)
MPMAF[%, MPExpandD, {At[2], {f, g}}]
```

$$y == \frac{\partial^3 (a f + b g)}{\partial x^2 \partial y}$$

$$y == a \frac{\partial^3 f}{\partial x^2 \partial y} + b \frac{\partial^3 g}{\partial x^2 \partial y}$$

The function **MPExpandD** is used to expand an expression using the differential operators.

```
MPExpandD[∂/∂x (f + Sin[g])/h, {f, g, h}]
```

$$\frac{1}{h}\left(\frac{\partial f}{\partial x} + \text{Cos}[g] \frac{\partial g}{\partial x}\right) - \frac{f + \text{Sin}[g]}{h^2} \frac{\partial h}{\partial x}$$

• Integration by parts using the differential operator

```
MPIntegrateByPartsD[HF@∫f ∂g/∂x dx]
```

$$f g - \int g \frac{\partial f}{\partial x} dx$$

• Merging the differential operator

```
MPMergeD[a ∂f/∂x + b ∂g/∂x]
```

$$\frac{\partial (a f + b g)}{\partial x}$$

• Transformation of variables

• Transformation of variables

```
MPTransD[∂^2 φ / ∂ω^2, VT → {ω, λ, ω == 2 π c / λ}]
```

$$\frac{\lambda^3}{2\,c^2\,\pi^2}\,\frac{\partial\phi}{\partial\lambda}+\frac{\lambda^4}{4\,c^2\,\pi^2}\,\frac{\partial^2\phi}{\partial\lambda^2}$$

• Evaluation of vector operators

```
MPAFE[∇^2 ψ, MPEvalVecOps, {All, ψ == ψ[ρ, φ], Cylindrical[ρ, φ, z]}, Apply → Expand]
MPMAF[%, MPSimpD, {1/ρ ∂ψ/∂ρ + ∂^2 ψ/∂ρ^2, Mult → ρ}]
```

$$\nabla^2\psi == \frac{1}{\rho}\,\frac{\partial\psi}{\partial\rho}+\frac{\partial^2\psi}{\partial\rho^2}+\frac{1}{\rho^2}\,\frac{\partial^2\psi}{\partial\varphi^2}$$

$$\nabla^2\psi == \frac{1}{\rho^2}\,\frac{\partial^2\psi}{\partial\varphi^2}+\frac{1}{\rho}\,\frac{\partial}{\partial\rho}\left(\rho\,\frac{\partial\psi}{\partial\rho}\right)$$

```
MPAFE[∇^2 ψ, MPEvalVecOps, {All, ψ == ψ[x, y, z], Cartesian[x, y, z]}]
MPMAF[%, MPTransD, {At[2],
  VT → {{x, y, z}, {r, θ, φ}, {x == r Sin[θ] Cos[φ], y == r Sin[θ] Sin[φ], z == r Cos[θ]}},
  Apply → (Expand@PowerExpand@Simplify@# &)},
 MPSimpD, {{2/r ∂ψ/∂r + ∂^2 ψ/∂r^2}, {Cot[θ]/r^2 ∂ψ/∂θ + 1/r^2 ∂^2 ψ/∂θ^2}}]
```

$$\nabla^2\psi == \frac{\partial^2\psi}{\partial x^2}+\frac{\partial^2\psi}{\partial y^2}+\frac{\partial^2\psi}{\partial z^2}$$

$$\nabla^2\psi == \frac{2}{r}\,\frac{\partial\psi}{\partial r}+\frac{\text{Cot}[\theta]}{r^2}\,\frac{\partial\psi}{\partial\theta}+\frac{\partial^2\psi}{\partial r^2}+\frac{1}{r^2}\,\frac{\partial^2\psi}{\partial\theta^2}+\frac{\text{Csc}[\theta]^2}{r^2}\,\frac{\partial^2\psi}{\partial\varphi^2}$$

$$\nabla^2\psi == \frac{\text{Csc}[\theta]^2}{r^2}\,\frac{\partial^2\psi}{\partial\varphi^2}+\frac{1}{r}\,\frac{\partial^2(r\,\psi)}{\partial r^2}+\frac{\text{Csc}[\theta]}{r^2}\,\frac{\partial}{\partial\theta}\left(\text{Sin}[\theta]\,\frac{\partial\psi}{\partial\theta}\right)$$

Integration

• Wrap the integral with **HoldForm** (HF) to prevent evaluation.

– Without the wrap

```
F == ∫f[x] dx
```

$$F == \int f[x]\,dx$$

– With the wrap

```
F == HF@∫f dx
```

$$F == \int f\, dx$$

• Combining multiple integrals

```
(HF@∫f[x] dx) (HF@∫g[x] dx)
MPCombInt[%, RV → {x, x′}]
```

$$\left(\int f[x]\, dx\right) \int g[x]\, dx$$

$$\int\int f[x]\, g[x']\, dx'\, dx$$

• Convolution in Fourier transform

```
{f[x] == HF@∫F[k] e^(i k x) dk, g[x] == HF@∫G[k] e^(i k x) dk}
```

$$\left\{f[x] == \int F[k]\, e^{i\,k\,x}\, dk,\ g[x] == \int G[k]\, e^{i\,k\,x}\, dk\right\}$$

```
HF@∫F[k] G[k] e^(i k x) dk
MPMAF[%, RA, {All, {F[k] == 1/(2 π) ∫f[x] e^(-i k x) dx, G[k] == 1/(2 π) ∫g[x] e^(-i k x) dx}},
 MPCombInt, {All, RV → {x, x′, x″}},
 MPFourierToDiracDelta, {All, k},
 MPEvalIntDelta, {All, x″}]
```

$$\int F[k]\, G[k]\, e^{i\,k\,x}\, dk$$

$$\frac{1}{2\,\pi} \int \left(\int f[x'']\, g[x']\, \delta[x - x' - x'']\, dx'\right) dx''$$

$$\frac{1}{2\,\pi} \int f[x - x']\, g[x']\, dx'$$

• Transformation of the integration variables

```
MPTransInt[∫f dx , VT → {x, y, y == x^2}]
```

$$\int \frac{f}{2\,\sqrt{y}}\, dy$$

```
MPAFE[HF@∫_-∞^∞ 1/(x^2 + a^2) dx, MPEvalInt, Apply → PowerExpand]
```

$$\int_{-\infty}^{\infty} \frac{1}{x^2 + a^2}\, dx == \frac{\pi}{a}$$

```
MPAFE[HF@∫_-∞^∞ 1/(x^2 + a^2) dx, MPTransInt,
 {All, VT → {x, θ, x == a Tan[θ]}}, RA → Sign[a] == 1, PostApply → MPEvalInt]
```

$$\int_{-\infty}^{\infty} \frac{1}{x^2 + a^2}\, dx == \frac{\pi}{a}$$

```
HF@Integrate[f, x, y, z]
MPMAF[%, MPTransInt, {All,
  VT → {{x, y, z}, {r, θ, ϕ}, {x == r Sin[θ] Cos[ϕ], y == r Sin[θ] Sin[ϕ], z == r Cos[θ]}}},
 Apply → PowerExpand]
```

$$\int\int\int f\, dz\, dy\, dx$$

$$\int\int\int f\, r^2\, \text{Sin}[\theta]\, d\phi\, d\theta\, dr$$

• Replacement of the integration variable with others

```
MPTransInt[∫ f dx^3 , Weight → r^2 Sin[θ], RV → {x^3, {r, 0, ∞}, {θ, 0, π}, {ϕ, 0, 2 π}}]
```

$$\int_0^{\infty} \int_0^{\pi} \int_0^{2\pi} f\, r^2\, \text{Sin}[\theta]\, d\phi\, d\theta\, dr$$

• Merging multiple integrals

```
MPMergeInt[a ∫f dx + b ∫g dx + c ∫h dx]
```

$$\int (a\, f + b\, g + c\, h)\, dx$$

• Merging integration intervals

```
MPMergeIntInterval[∫_0^a f[t] dt + ∫_a^b f[t] dt]
```

$$\int_0^b f[t]\, dt$$

• Integration by parts

```
MPIntegrateByParts[HF@∫_-∞^x ψ'[y] y^2 dy, ψ'[y]]
```

$$-\int_{-\infty}^{x} 2\, y\, \psi[y]\, dy - \left(y^2\, \psi[y]\right)_{y==-\infty} + x^2\, \psi[x]$$

Summation

• Changing the summation intervals

```
MPChSumInterval[HF@ Sum[a_n, {n, {1, 2}, {n_3, n_4}}]]
```

$$\sum_{n=1}^{2} a_n + \sum_{n=n_3}^{n_4} a_n$$

• Combining multiple summations. The indexes are replaced to avoid duplication.

```
MPCombSum[Sum[f_n, {n, 1, ∞}] Sum[g_n, {n, 1, ∞}], RV → {n, m}]
```

$$\sum_{n=1}^{\infty} \sum_{m=1}^{\infty} f_n \, g_m$$

• Merging multiple summations

```
MPMergeSum[Sum[A[k], {k, 0, ∞}] + Sum[B[k], {k, 0, ∞}]]
```

$$\sum_{k=0}^{\infty} (A[k] + B[k])$$

• Merging summation intervals

```
MPMergeSumInterval[Sum[a_n, {n, 0, k-1}] + Sum[a_n, {n, k, N}]]
```

$$\sum_{n=0}^{N} a_n$$

Vectors and Matrices

• Expansion of vector expressions

```
MPExpandVec[A⃗ × (a B⃗ × C⃗)]
```

$$a\,\vec{A}.\vec{C}\,\vec{B} - a\,\vec{A}.\vec{B}\,\vec{C}$$

```
MPExpandVec[∇ · (ϵ E⃗), {ϵ, E⃗}]
```

$$\epsilon\, \nabla \cdot \vec{E} + (\nabla \epsilon).\vec{E}$$

• Matrix form of equations

```
MPEqsMatForm[{a x + b y == z_1, c x + d y == z_2}, {x, y}]
```

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix} \cdot \begin{pmatrix} x \\ y \end{pmatrix} == \begin{pmatrix} z_1 \\ z_2 \end{pmatrix}$$

• Expansion of $\nabla \times \nabla \times \nabla \mathbf{A}$

• Expansion of $\nabla \times \nabla \times \nabla \mathbf{A}$

```
MPExpandVec[∇ × ∇ × A⃗, A⃗]
```

$\nabla\left(\nabla \cdot \vec{A}\right) - \nabla^2 \vec{A}$

• Deleting a column from a matrix

```
Table[i j, {i, 3}, {j, 3}] // MatrixForm
MPDeleteColumns[%, 2] // MatrixForm
```

$$\begin{pmatrix} 1 & 2 & 3 \\ 2 & 4 & 6 \\ 3 & 6 & 9 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 3 \\ 2 & 6 \\ 3 & 9 \end{pmatrix}$$

• Inserting a column in a matrix

```
Table[i j, {i, 3}, {j, 3}] // MatrixForm
MPInsertColumns[%, {0, 0, 0}, -1] // MatrixForm
```

$$\begin{pmatrix} 1 & 2 & 3 \\ 2 & 4 & 6 \\ 3 & 6 & 9 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 2 & 3 & 0 \\ 2 & 4 & 6 & 0 \\ 3 & 6 & 9 & 0 \end{pmatrix}$$

• Diagonalization of a matrix

```
{{1, 4, 9}, {4, 4, 0}, {9, 0, 4}};
MPDiagMatrix[%] // MatrixForm
```

$$\begin{pmatrix} \frac{1}{2}\left(5+\sqrt{397}\right) & 0 & 0 \\ 0 & \frac{1}{2}\left(5-\sqrt{397}\right) & 0 \\ 0 & 0 & 4 \end{pmatrix}$$

• Transform a vector from Cartesian to spherical coordinates

```
MPVecFromCart[{0, 0, 1}, Spherical[r, θ, φ]]
```

{Cos[θ], -Sin[θ], 0}

The result is $\hat{\mathbf{z}} = \hat{\mathbf{r}}\cos\theta - \hat{\boldsymbol{\theta}}\sin\theta$.

• Transform a vector from spherical to Cartesian coordinates

```
MPVecToCart[{0, 0, 1}, Spherical[r, θ, φ]]
```

```
{-Sin[φ], Cos[φ], 0}
```

The result is $\hat{\boldsymbol{\varphi}} = -\hat{\boldsymbol{x}}\sin\varphi + \hat{\mathbf{y}}\cos\varphi$.

Functional Analysis

• Variable transformation

```
y'[x] + y[x]
MPTransEq[%, y[x] == g[s], VT → {x, s, x == s^2, s == √x}]
```

```
y[x] + y'[x]
```

$$g[s] + \frac{g'[s]}{2\,s}$$

A case of two variables

```
∂x,x f[x, y] - ∂y,y f[x, y]
MPTransEq[%, f[x, y] == g[ξ, η],
 VT → {{x, y}, {ξ, η}, {ξ == (x + y)/2, η == (x - y)/2}}, Apply → Simplify]
```

$$-f^{(0,2)}[x, y] + f^{(2,0)}[x, y]$$

$$g^{(1,1)}[\xi, \eta]$$

• Abbreviation of derivatives

```
MPAbbrevF[i p^(0,1)[x, t] - 2 α p^(1,0)[x, t] q^(1,0)[x, t], Deriv → True]
```

$$i\,p_t - 2\,\alpha\,p_x\,q_x$$

The subscripts denote the derivatives. The function arguments can be restored using `MPRestoreFunctions`.

```
MPRestoreF[i p_t - 2 α p_x q_x, {p, q}, Deriv → True]
```

$$-2\,\alpha\,p^{(0,1)}[t, x]\,q^{(0,1)}[t, x] + i\,p^{(1,0)}[t, x]$$

• Taking a factor out of a function

```
MPFactorFunc[f[a x] + ∇ × (a A), CurlH, a]
```

$$a\,\nabla\times A + f[a\,x]$$

• Putting a factor inside a function

```
MPInsideFunc[f[a x] + a ∇ × A, CurlH, a]
```

```
∇ × (a A) + f[a x]
```

• Merging arguments of a linear function

```
MPMergeFunc[a f[x, z] + b f[y, z], f]
```

```
f[a x + b y, z]
```

Operator Analysis

• Expansion of operator expressions

```
MPExpandBraket[(⟨a| + ⟨b|) ∘ A ∘ (|α⟩ + |β⟩), A]
```

```
⟨a | A | α⟩ + ⟨a | A | β⟩ + ⟨b | A | α⟩ + ⟨b | A | β⟩
```

```
MPCommutator[Q/(λ0 √2) + i/(ħ √2) λ0 P, Q/(λ0 √2) - i/(ħ √2) λ0 P]
MPExpandOp[%, {Q, P}]
```

$$-\left(\frac{Q}{\sqrt{2}\ \lambda_0}-\frac{i\ P\ \lambda_0}{\sqrt{2}\ \hbar}\right)\circ\left(\frac{Q}{\sqrt{2}\ \lambda_0}+\frac{i\ P\ \lambda_0}{\sqrt{2}\ \hbar}\right)+\left(\frac{Q}{\sqrt{2}\ \lambda_0}+\frac{i\ P\ \lambda_0}{\sqrt{2}\ \hbar}\right)\circ\left(\frac{Q}{\sqrt{2}\ \lambda_0}-\frac{i\ P\ \lambda_0}{\sqrt{2}\ \hbar}\right)$$

$$\frac{i}{\hbar}\ P\circ Q-\frac{i}{\hbar}\ Q\circ P$$

• Simplify a commutator using a commutation rule $[Q, P] \equiv QP - PQ = i\hbar$

```
MPCommutator[Q^2, P^2]
MPSimpOp[%, {Q, P, i ħ}]
```

$$-P^2\circ Q^2+Q^2\circ P^2$$

$$2\hbar^2+4\ i\ \hbar\ Q\circ P$$

Examples

– Ref. Chap 9, Arfken, Mathematical Methods for Physicists

■ Parachutist

• The equation of motion is

```
m v̇ == m g - b v^2
```

where $-b\,v^2$ is the drag force and $m\,g$ is the force of the gravitational attraction. The initial velocity when the parachute opens at time $t = 0$ is $v(0) = v_i$.

• The terminal velocity, v_0, can be found from the equation of motion as $t \to \infty$; when there is no acceleration, $\dot{v} = 0$, so

• The terminal velocity, v_0, can be found from the equation of motion as $t \to \infty$; when there is no acceleration, $\dot{v} = 0$, so

```
m v̇ == m g - b v^2
MPMAF[%, RA, {All, {v̇ == 0, v == v0}},
 MPSolve, {All, v0, -1}, MergeP → True]
```

$$m\,\dot{v} == g\,m - b\,v^2$$

$$0 == g\,m - b\,v_0^2$$

$$v_0 == \sqrt{\frac{g\,m}{b}}$$

• The variables t and v separate.

```
m v̇ == m g - b v^2
MPMAF[%, RA, {At[1], v̇ == dv/dt},
 MPSepVars, {All, t, Times, Side → Right},
 MPDivDenom, {At[1], m}]
```

$$m\,\dot{v} == g\,m - b\,v^2$$

$$\frac{m\,dv}{g\,m - b\,v^2} == dt$$

$$\frac{dv}{g - \frac{b\,v^2}{m}} == dt$$

which can be integrated to yield the velocity at time t.

```
dv/(g - b v^2/m) == dt
MPMAF[%, MPDiffToInt, {All, {v, vi, v}, t}, ,
 MPEvalInt, All, , ,
 MPSolve, {All, v}, Hold → vi, MergeP → True, ,
 TrigExpand, At[2], Apply → Simplify,
 MPDivFrac, {At[2], Sqrt[b/(g m)] Cosh[Sqrt[b g/m] t]}, MergeP → True, ,
 MPEli, {At[2], v0 == Sqrt[g m/b], b}, Apply → PowerExpand, , ,
 MPEli, {At[2], v0/g == T, g}]
```

$$\frac{dv}{g - \frac{b\,v^2}{m}} == dt$$

$$\frac{dv}{g - \frac{b\,v^2}{m}} == dt$$

$$\int_{v_i}^{v} \frac{1}{g - \frac{b\,v^2}{m}}\,dv == \int 1\,dt$$

$$v == \sqrt{\frac{g\,m}{b}}\ \text{Tanh}\left[\sqrt{\frac{b\,g}{m}}\ t + \text{ArcTanh}\left[\sqrt{\frac{b}{g\,m}}\ v_i\right]\right]$$

$$v == \sqrt{\frac{g\,m}{b}}\ \frac{v_i + \sqrt{\frac{g\,m}{b}}\ \text{Tanh}\left[\sqrt{\frac{b\,g}{m}}\ t\right]}{\sqrt{\frac{g\,m}{b}} + v_i\ \text{Tanh}\left[\sqrt{\frac{b\,g}{m}}\ t\right]}$$

$$v == \frac{v_0\ \left(v_i + v_0\ \text{Tanh}\left[\frac{t}{T}\right]\right)}{v_0 + v_i\ \text{Tanh}\left[\frac{t}{T}\right]}$$

where

$$\left\{v_0 == \sqrt{\frac{g\,m}{b}}\ ,\ T == \frac{v_0}{g}\right\}$$

• Verify that our solution satisfies the originial equation of motion:

```
MPAFE[v̇, MPExpandD, {All, v == v0 (vi + v0 Tanh[t/T])/(v0 + vi Tanh[t/T])}, OverDot → t}, Apply → Simplify]
MPMAF[%, MPTrigConvert, {At[2], Tanh},
 MPEli, {At[2], v == v0 (vi + v0 Tanh[t/T])/(v0 + vi Tanh[t/T]), Tanh[t/T]}, Apply → Simplify,
 RR, {At[2], {v0 == Sqrt[g m/b], T == v0/g}}, Apply → Expand]
```

$$\dot{v} == \frac{v_0\ \left(v_0^2 - v_i^2\right)}{T\ \left(v_0\ \text{Cosh}\left[\frac{t}{T}\right] + v_i\ \text{Sinh}\left[\frac{t}{T}\right]\right)^2}$$

$$\dot{v} == \frac{-v^2 + v_0^2}{T\ v_0}$$

$$\dot{v} == g - \frac{b\,v^2}{m}$$

that is, Newton's equation of motion.

• Substituting the numerical data, v_0 and T are

```
{v0 == Sqrt[g m / b], T == v0 / g}

MPMAF[%, MPEli, {All, v0, 1, Keep → True}, ExpandMergeP → True,
 RA, {All, {g == 9.8 Meter / Second^2, b == 700 Kilo Gram / Meter, m == 70 Kilo Gram}}, Apply → PowerExpand,
 Convert, {At[1, 2], Mile / Hour}]
```

$$\left\{v_0 == \sqrt{\frac{g\,m}{b}},\ T == \frac{v_0}{g}\right\}$$

$$\left\{v_0 == \frac{0.989949\ \text{Meter}}{\text{Second}},\ T == 0.101015\ \text{Second}\right\}$$

$$\left\{v_0 == \frac{2.21445\ \text{Mile}}{\text{Hour}},\ T == 0.101015\ \text{Second}\right\}$$

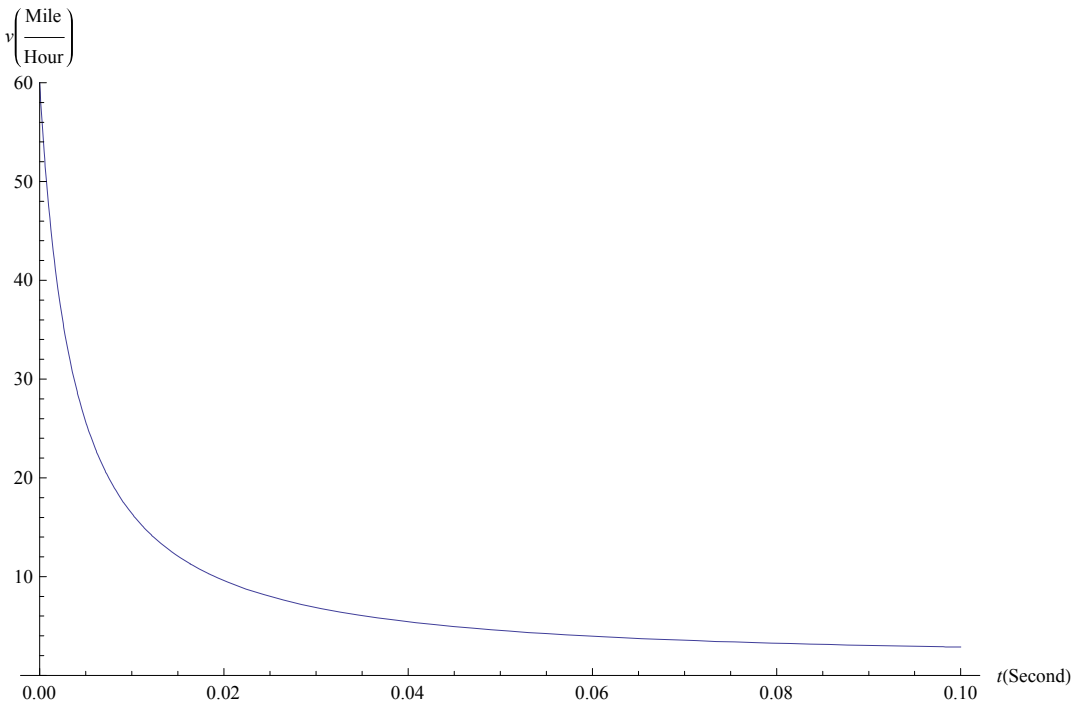
Plot $v(t)$.

```
v == Sqrt[g m / b] (v_i + Sqrt[g m / b] Tanh[Sqrt[b g / m] t]) / (Sqrt[g m / b] + v_i Tanh[Sqrt[b g / m] t])

MPMAF[%, RA, {At[2], {g == 9.8 Meter / Second^2, b == 700 Kilo Gram / Meter,
    m == 70 Kilo Gram, v_i == 60 Mile / Hour, t == t Second}}, Apply → PowerExpand,
  MPDivFrac, {At[2], Mile / Hour},
  Convert, {{Hour Meter / (Mile Second), 1}, {At[2], Mile / Hour}},
  MPDivEq, {All, Mile / Hour},
  Plot, {At[2], {t, 0, 0.1}, PlotRange → {0, 60},
   AxesLabel → {t[Second], v[Mile / Hour]}, ImageSize → 500}, Take → Last]
```

$$v == \sqrt{\frac{g\,m}{b}}\,\frac{v_i + \sqrt{\frac{g\,m}{b}}\,\text{Tanh}\left[\sqrt{\frac{b\,g}{m}}\,t\right]}{\sqrt{\frac{g\,m}{b}} + v_i\,\text{Tanh}\left[\sqrt{\frac{b\,g}{m}}\,t\right]}$$

$$\frac{\text{Hour}\,v}{\text{Mile}} == \frac{2.21445\,(60 + 2.21445\,\text{Tanh}[9.89949\,t])}{2.21445 + 60\,\text{Tanh}[9.89949\,t]}$$



■ Series Solution of Differential Equations

• We apply the method of series soution to the linear (classical) oscillator equation.

(9.84)

$$\frac{d^2 y}{dx^2} + \omega^2 y == 0$$

with known solutions $y = \sin(\omega x),\ \cos(\omega x)$.

– We try the series solution of the form

(9.85)

$$y[x] == x^k \left(a_0 + a_1 x + a_2 x^2 + a_3 x^3 + \cdots\right) == HF@\sum_{\lambda=0}^{\infty} a_\lambda x^{k+\lambda}$$

$$y[x] == x^k \left(\cdots + a_0 + x\, a_1 + x^2\, a_2 + x^3\, a_3\right) == \sum_{\lambda=0}^{\infty} a_\lambda x^{k+\lambda}$$

with $a_0 \neq 0$ and the exponent k and all the coefficients a_λ still undetermined.

– By substituting (9.85) into Eq. (9.84), we have

(9.86)

$$\frac{d^2 y}{dx^2} + \omega^2 y == 0$$

$$MPMAF\left[\%, RA, \left\{At[1], y == \sum_{\lambda=0}^{\infty} a_\lambda x^{k+\lambda}\right\}, Apply \to MPEvalD\right]$$

$$y\,\omega^2 + \frac{d^2 y}{dx^2} == 0$$

$$\omega^2 \sum_{\lambda=0}^{\infty} x^{k+\lambda}\, a_\lambda + \sum_{\lambda=0}^{\infty} x^{-2+k+\lambda}\, (-1 + k + \lambda)\, (k + \lambda)\, a_\lambda == 0$$

which gives

```
ω^2 Sum[x^(k+λ) a_λ, {λ, 0, ∞}] + Sum[x^(-2+k+λ) (-1 + k + λ) (k + λ) a_λ, {λ, 0, ∞}] == 0
MPMAF[%, MPShiftSum, {At[1, x^(-2+k+λ)], {λ, 2}},
 MPChSumLimits, {At[1], {λ, 0, ∞}},
 MPMergeSum, At[1],
 MPFactor, {At[1], x^(k+λ)}, Base → _Sum]
```

$$\omega^2 \sum_{\lambda=0}^{\infty} x^{k+\lambda}\, a_\lambda + \sum_{\lambda=0}^{\infty} x^{-2+k+\lambda}\, (-1+k+\lambda)\, (k+\lambda)\, a_\lambda == 0$$

$$\sum_{\lambda=0}^{\infty} \left(x^{k+\lambda}\, \omega^2\, a_\lambda + x^{k+\lambda}\, (1+k+\lambda)\, (2+k+\lambda)\, a_{2+\lambda}\right) + (-1+k)\, k\, x^{-2+k}\, a_0 + k\, (1+k)\, x^{-1+k}\, a_1 == 0$$

$$\sum_{\lambda=0}^{\infty} x^{k+\lambda} \left(\omega^2\, a_\lambda + (1+k+\lambda)\, (2+k+\lambda)\, a_{2+\lambda}\right) + (-1+k)\, k\, x^{-2+k}\, a_0 + k\, (1+k)\, x^{-1+k}\, a_1 == 0$$

– From the lowest power of x, x^{k-2}, vanishing of the coefficient yields

```
(-1 + k) k a_0 == 0
```

Since a_0 was chosen to be nonzero, we have

(9.87)

```
(-1 + k) k == 0
```

This equation is called the indicial equation and we have $k = 0$ or $k = 1$.

– Since the coefficient of $x^{k+j} (j \geq 0)$ vanishes, this results in

(9.88)

```
ω^2 a_λ + (1 + k + λ) (2 + k + λ) a_(2+λ) == 0
MPMAF[%, RA, {At[1], λ == j},
 MPSolve, {All, a_(2+j)}]
```

$$\omega^2\, a_\lambda + (1+k+\lambda)\, (2+k+\lambda)\, a_{2+\lambda} == 0$$

$$\omega^2\, a_j + (1+j+k)\, (2+j+k)\, a_{2+j} == 0$$

$$a_{2+j} == -\frac{\omega^2\, a_j}{(1+j+k)\, (2+j+k)}$$

• Depending on whether $k = 0$ or $k = 1$, we have different recurrence relations. First with $k = 0$, the recurrence relation (Eq. (9.88)) becomes

(9.89)

```
a_{2+j} == -ω^2 a_j / ((1 + j + k) (2 + j + k))
MPMAF[%, RA, {At[2], k == 0}]
```

$$a_{2+j} == -\frac{\omega^2\, a_j}{(1 + j + k)\,(2 + j + k)}$$

$$a_{2+j} == -\frac{\omega^2\, a_j}{(1 + j)\,(2 + j)}$$

whose solution is

(9.90)

```
a_{2+j} == -ω^2 a_j / ((1 + j) (2 + j))
MPMAF[%, MPRSolve, {All, a_j, {j, 0}},
 RA, {All, j == 2 n}, SimpOne → True, Apply → PowerExpand,
 MPToFactorial, At[2]]
```

$$a_{2+j} == -\frac{\omega^2\, a_j}{(1 + j)\,(2 + j)}$$

$$a_{2\,n} == \frac{(-1)^n\, \omega^{2\,n}\, a_0}{\text{Gamma}[1 + 2\,n]}$$

$$a_{2\,n} == \frac{(-1)^n\, \omega^{2\,n}\, a_0}{(2\,n)\,!}$$

and our solution of the differential equation is

(9.91)

```
y == Sum[a_λ x^(k+λ), {λ, 0, ∞}]
MPMAF[%, RA, {At[2], k == 0},
 MPChSumInterval, {At[2], {λ, {n, 0, ∞, λ == 2 n}}},
 RA, {At[2], a_{2 n} == (-1)^n ω^(2 n) a_0 / (2 n)!}, Apply → MPFactorSum,
 MPEvalSum, At[2]]
```

$$y == \sum_{\lambda=0}^{\infty} a_\lambda\, x^{k+\lambda}$$

$$y == a_0 \sum_{n=0}^{\infty} \frac{(-1)^n\, x^{2\,n}\, \omega^{2\,n}}{(2\,n)\,!}$$

$$y == \text{Cos}[x\,\omega]\, a_0$$

• Similarly, if we choose the indicial equation root $k = 1$ (Eq. (9.88)), the solution becomes

(9.94)

$$y == \frac{\text{Sin}[x\,\omega]\,a_0}{\omega}$$

• From (9.91) and (9.94), the general solution can be put

$$y == c_1\,\text{Cos}[x\,\omega] + c_2\,\text{Sin}[x\,\omega]$$

Mathematica Language

- **Language Overview (*Mathematica* Guide)**

Structure of *Mathematica* Expressions

- **Expressions (*Mathematica* Tutorial)**

Constants

- **Numbers (*Mathematica* Overview)**
- **Types of Numbers (*Mathematica* Tutorial)**
- **Integer (Built-in *Mathematica* Symbol)**
- **Rational (Built-in *Mathematica* Symbol)**
- **Real (Built-in *Mathematica* Symbol)**
- **Complex (Built-in *Mathematica* Symbol)**
- **Mathematical Constants (*Mathematica* Guide)**

Variables

- **Defining Variables (*Mathematica* Tutorial)**
- **Eliminating Variables (*Mathematica* Tutorial)**

Patterns

- **Patterns (*Mathematica* Guide)**
- **Introduction to Patterns**
- **Verbatim Patterns**
- **Example**

• In four-wave mixing, two waves of frequencies ω_1 and ω_2 interact and generate additional frequencies through nonlinear mixing. Suppose the two waves are given by

```
E_i == 1/2 (Ẽ_i e^(-i ω_i t) + Ẽ_i^* e^(i ω_i t))
```

for $i = 1, 2$, and the nonlinear process is due to the third-order Kerr effect:

```
P_NL == ϵ_0 χ_3 E^3
```

Find out what frequencies are present in the output wave near $\omega_1 \approx \omega_2$.

• The field is given by

```
E == HF@Sum[E_i, {i, 1, 2}]
MPMAF[%, RA, {At[2], E_i == 1/2 (Ẽ_i e^(-i ω_i t) + Ẽ_i^* e^(i ω_i t))},
 MPEvalSum, At[2]]
```

$$E == \sum_{i=1}^{2} E_i$$

$$E == \frac{1}{2} \sum_{i=1}^{2} \left(e^{-i t \omega_i} \tilde{E}_i + e^{i t \omega_i} (\tilde{E}_i)^*\right)$$

$$E == \frac{1}{2} \left(e^{-i t \omega_1} \tilde{E}_1 + e^{-i t \omega_2} \tilde{E}_2 + e^{i t \omega_1} (\tilde{E}_1)^* + e^{i t \omega_2} (\tilde{E}_2)^*\right)$$

and the nonlinear mixing produces

```
P_NL == ϵ_0 χ_3 E^3
MPMAF[%, RA, {At[2], E == 1/2 (e^(-i t ω_1) Ẽ_1 + e^(-i t ω_2) Ẽ_2 + e^(i t ω_1) (Ẽ_1)^* + e^(i t ω_2) (Ẽ_2)^*)},
 Expand, At[2, E], FactorExp → i t]
```

$$P_{NL} == E^3 \epsilon_0 \chi_3$$

$$P_{NL} == \frac{1}{8} \epsilon_0 \chi_3 \left(e^{-i t \omega_1} \tilde{E}_1 + e^{-i t \omega_2} \tilde{E}_2 + e^{i t \omega_1} (\tilde{E}_1)^* + e^{i t \omega_2} (\tilde{E}_2)^*\right)^3$$

$$\begin{aligned} P_{NL} == \frac{1}{8} \epsilon_0 \chi_3 \Big(& e^{-3 i t \omega_1} \tilde{E}_1^3 + 3 e^{i t (-2\omega_1 - \omega_2)} \tilde{E}_1^2 \tilde{E}_2 + 3 e^{i t (-\omega_1 - 2\omega_2)} \tilde{E}_1 \tilde{E}_2^2 + e^{-3 i t \omega_2} \tilde{E}_2^3 + 3 e^{-i t \omega_1} \tilde{E}_1^2 (\tilde{E}_1)^* + \\ & 6 e^{-i t \omega_2} \tilde{E}_1 \tilde{E}_2 (\tilde{E}_1)^* + 3 e^{i t (\omega_1 - 2\omega_2)} \tilde{E}_2^2 (\tilde{E}_1)^* + 3 e^{i t \omega_1} \tilde{E}_1 \left((\tilde{E}_1)^*\right)^2 + 3 e^{i t (2\omega_1 - \omega_2)} \tilde{E}_2 \left((\tilde{E}_1)^*\right)^2 + \\ & e^{3 i t \omega_1} \left((\tilde{E}_1)^*\right)^3 + 3 e^{i t (-2\omega_1 + \omega_2)} \tilde{E}_1^2 (\tilde{E}_2)^* + 6 e^{-i t \omega_1} \tilde{E}_1 \tilde{E}_2 (\tilde{E}_2)^* + 3 e^{-i t \omega_2} \tilde{E}_2^2 (\tilde{E}_2)^* + \\ & 6 e^{i t \omega_2} \tilde{E}_1 (\tilde{E}_1)^* (\tilde{E}_2)^* + 6 e^{i t \omega_1} \tilde{E}_2 (\tilde{E}_1)^* (\tilde{E}_2)^* + 3 e^{i t (2\omega_1 + \omega_2)} \left((\tilde{E}_1)^*\right)^2 (\tilde{E}_2)^* + \\ & 3 e^{i t (-\omega_1 + 2\omega_2)} \tilde{E}_1 \left((\tilde{E}_2)^*\right)^2 + 3 e^{i t \omega_2} \tilde{E}_2 \left((\tilde{E}_2)^*\right)^2 + 3 e^{i t (\omega_1 + 2\omega_2)} (\tilde{E}_1)^* \left((\tilde{E}_2)^*\right)^2 + e^{3 i t \omega_2} \left((\tilde{E}_2)^*\right)^3 \Big) \end{aligned}$$

The frequencies are

```
PNL == 1/8 ϵ0 χ3 (e^(-3 i t ω1) Ẽ1^3 + 3 e^(i t (-2 ω1-ω2)) Ẽ1^2 Ẽ2 + 3 e^(i t (-ω1-2 ω2)) Ẽ1 Ẽ2^2 + e^(-3 i t ω2) Ẽ2^3 + 3 e^(-i t ω1) Ẽ1^2 (Ẽ1)* +
    6 e^(-i t ω2) Ẽ1 Ẽ2 (Ẽ1)* + 3 e^(i t (ω1-2 ω2)) Ẽ2^2 (Ẽ1)* + 3 e^(i t ω1) Ẽ1 ((Ẽ1)*)^2 + 3 e^(i t (2 ω1-ω2)) Ẽ2 ((Ẽ1)*)^2 +
    e^(3 i t ω1) ((Ẽ1)*)^3 + 3 e^(i t (-2 ω1+ω2)) Ẽ1^2 (Ẽ2)* + 6 e^(-i t ω1) Ẽ1 Ẽ2 (Ẽ2)* + 3 e^(-i t ω2) Ẽ2^2 (Ẽ2)* +
    6 e^(i t ω2) Ẽ1 (Ẽ1)* (Ẽ2)* + 6 e^(i t ω1) Ẽ2 (Ẽ1)* (Ẽ2)* + 3 e^(i t (2 ω1+ω2)) ((Ẽ1)*)^2 (Ẽ2)* +
    3 e^(i t (-ω1+2 ω2)) Ẽ1 ((Ẽ2)*)^2 + 3 e^(i t ω2) Ẽ2 ((Ẽ2)*)^2 + 3 e^(i t (ω1+2 ω2)) (Ẽ1)* ((Ẽ2)*)^2 + e^(3 i t ω2) ((Ẽ2)*)^3)
MPMAF[%, frequencies == Cases[#, e^a_, {0, ∞}] &, At[2], Take → Last,
 Part, {At[2], {2}}, Level → 1,
 MPDiv, {At[2], i t},
 Cases, {At[2], ωi_ | 2 ωi_ - ωj_}, Apply → Union]
```

$$P_{NL} == \frac{1}{8}\epsilon_0 \chi_3 \left(e^{-3 i t \omega_1} \tilde{E}_1^3 + 3 e^{i t (-2\omega_1-\omega_2)} \tilde{E}_1^2 \tilde{E}_2 + 3 e^{i t (-\omega_1-2\omega_2)} \tilde{E}_1 \tilde{E}_2^2 + e^{-3 i t \omega_2} \tilde{E}_2^3 + 3 e^{-i t \omega_1} \tilde{E}_1^2 (\tilde{E}_1)^* + 6 e^{-i t \omega_2} \tilde{E}_1 \tilde{E}_2 (\tilde{E}_1)^* + 3 e^{i t (\omega_1-2\omega_2)} \tilde{E}_2^2 (\tilde{E}_1)^* + 3 e^{i t \omega_1} \tilde{E}_1 ((\tilde{E}_1)^*)^2 + 3 e^{i t (2\omega_1-\omega_2)} \tilde{E}_2 ((\tilde{E}_1)^*)^2 + e^{3 i t \omega_1} ((\tilde{E}_1)^*)^3 + 3 e^{i t (-2\omega_1+\omega_2)} \tilde{E}_1^2 (\tilde{E}_2)^* + 6 e^{-i t \omega_1} \tilde{E}_1 \tilde{E}_2 (\tilde{E}_2)^* + 3 e^{-i t \omega_2} \tilde{E}_2^2 (\tilde{E}_2)^* + 6 e^{i t \omega_2} \tilde{E}_1 (\tilde{E}_1)^* (\tilde{E}_2)^* + 6 e^{i t \omega_1} \tilde{E}_2 (\tilde{E}_1)^* (\tilde{E}_2)^* + 3 e^{i t (2\omega_1+\omega_2)} ((\tilde{E}_1)^*)^2 (\tilde{E}_2)^* + 3 e^{i t (-\omega_1+2\omega_2)} \tilde{E}_1 ((\tilde{E}_2)^*)^2 + 3 e^{i t \omega_2} \tilde{E}_2 ((\tilde{E}_2)^*)^2 + 3 e^{i t (\omega_1+2\omega_2)} (\tilde{E}_1)^* ((\tilde{E}_2)^*)^2 + e^{3 i t \omega_2} ((\tilde{E}_2)^*)^3\right)$$

```
frequencies == {-3 ω1, -2 ω1 - ω2, -ω1 - 2 ω2, -3 ω2, -ω1, -ω2, ω1 - 2 ω2, ω1,
  2 ω1 - ω2, 3 ω1, -2 ω1 + ω2, -ω1, -ω2, ω2, ω1, 2 ω1 + ω2, -ω1 + 2 ω2, ω2, ω1 + 2 ω2, 3 ω2}
    frequencies == {ω1, 2 ω1 - ω2, ω2, -ω1 + 2 ω2}
```

Four frequencies are present in the output wave and they are in the order

```
frequencies == {2 ω1 - ω2, ω1, ω2, -ω1 + 2 ω2}
```

if $\omega_1 < \omega_2$.

Functions

- **Mathematical Functions (*Mathematica* Guide)**

Programming

- **Procedural Programming (*Mathematica* Guide)**
- **Defining Functions (*Mathematica* Tutorial)**

• Functions can be called recursively.

```
f[n_] := n f[n - 1]; f[0] = 1;
```

```
f[100]
```

```
93 326 215 443 944 152 681 699 238 856 266 700 490 715 968 264 381 621 468 592 963 895 217 599 993 229 ⁖
 915 608 941 463 976 156 518 286 253 697 920 827 223 758 251 185 210 916 864 000 000 000 000 000 000 ⁖
 000 000
```

```
100 !
```

```
93 326 215 443 944 152 681 699 238 856 266 700 490 715 968 264 381 621 468 592 963 895 217 599 993 229 ⁖
 915 608 941 463 976 156 518 286 253 697 920 827 223 758 251 185 210 916 864 000 000 000 000 000 000 ⁖
 000 000
```

```
Clear[f]
```

- **Functional Programming (*Mathematica* Guide)**
- **Pure Functions (*Mathematica* Tutorial)**
- **Functional Operations (*Mathematica* Overview)**

- **Example: data manipulation**

– Generate the data

```
data = RandomReal[{-10, 10}, 100]
```

```
{9.2069, 4.38196, 7.36707, 2.44153, -2.71293, -7.84716, -2.51371, -7.20865, -0.35902,
 0.547298, -5.73894, 6.72848, 0.891682, 6.47749, 2.32973, -1.13296, -9.28741, 2.6097,
 -3.3001, 1.31403, 3.28091, -6.56867, -4.45742, -3.50691, -5.22019, -7.31909,
 8.72281, 2.20909, -1.75738, 8.50959, 4.91354, 7.45721, 5.48025, -1.42342, 9.46615,
 4.72104, 7.33425, -1.08151, 5.55214, 2.43006, 8.78305, -8.43983, -1.63389,
 -5.64446, -6.33726, 6.14862, -5.05503, 7.83732, 9.89825, -6.70792, -1.25488,
 5.78948, -6.64355, -4.31986, 5.54391, 5.09748, 4.97521, -7.33067, -5.96476,
 0.273494, 4.95501, -9.22198, -4.24909, -6.01423, 1.07997, -0.97207, -0.425487,
 1.10524, 4.61603, -1.08979, -4.19288, -4.14596, 0.206045, -3.44104, -5.51746,
 -3.39021, 4.92202, -8.64063, -2.70173, 2.66111, 7.56427, 3.89614, 4.80096,
 -7.25148, -9.66792, 2.19267, -8.06609, 2.25114, 5.52372, 9.20761, -7.88423,
 4.23404, -6.21864, 6.96702, 5.57343, 7.28515, 9.36244, -4.81503, 6.72125, -0.399189}
```

– Number of data

```
Length[data]
```

```
100
```

– Addition of all data

```
Plus @@ data
```

```
34.7723
```

– Select positive numbers

```
Select[data, Negative]
```

```
{-2.71293, -7.84716, -2.51371, -7.20865, -0.35902, -5.73894, -1.13296, -9.28741,
 -3.3001, -6.56867, -4.45742, -3.50691, -5.22019, -7.31909, -1.75738, -1.42342,
 -1.08151, -8.43983, -1.63389, -5.64446, -6.33726, -5.05503, -6.70792, -1.25488,
 -6.64355, -4.31986, -7.33067, -5.96476, -9.22198, -4.24909, -6.01423, -0.97207,
 -0.425487, -1.08979, -4.19288, -4.14596, -3.44104, -5.51746, -3.39021, -8.64063,
 -2.70173, -7.25148, -9.66792, -8.06609, -7.88423, -6.21864, -4.81503, -0.399189}
```

```
Cases[data, _?Negative]
```

```
{-2.71293, -7.84716, -2.51371, -7.20865, -0.35902, -5.73894, -1.13296, -9.28741,
 -3.3001, -6.56867, -4.45742, -3.50691, -5.22019, -7.31909, -1.75738, -1.42342,
 -1.08151, -8.43983, -1.63389, -5.64446, -6.33726, -5.05503, -6.70792, -1.25488,
 -6.64355, -4.31986, -7.33067, -5.96476, -9.22198, -4.24909, -6.01423, -0.97207,
 -0.425487, -1.08979, -4.19288, -4.14596, -3.44104, -5.51746, -3.39021, -8.64063,
 -2.70173, -7.25148, -9.66792, -8.06609, -7.88423, -6.21864, -4.81503, -0.399189}
```

– Select data larger than 5

```
Select[data, # > 5 &]
```

```
{7.97756, 7.06071, 6.25687, 7.77989, 8.95827, 9.383, 6.92007, 6.09351,
 9.35384, 8.79465, 6.81701, 9.68147, 5.23082, 6.14421, 8.4387, 6.12878,
 7.46329, 6.51098, 9.32175, 8.12827, 9.10668, 9.73478, 8.44398, 6.23945}
```

```
Cases[data, _?(# > 5 &)]
```

```
{7.97756, 7.06071, 6.25687, 7.77989, 8.95827, 9.383, 6.92007, 6.09351,
 9.35384, 8.79465, 6.81701, 9.68147, 5.23082, 6.14421, 8.4387, 6.12878,
 7.46329, 6.51098, 9.32175, 8.12827, 9.10668, 9.73478, 8.44398, 6.23945}
```

– Group two numbers and add the numbers in each group.

```
Partition[#, 2] &@data
Plus @@@ %
```

```
{{9.2069, 4.38196}, {7.36707, 2.44153}, {-2.71293, -7.84716}, {-2.51371, -7.20865},
 {-0.35902, 0.547298}, {-5.73894, 6.72848}, {0.891682, 6.47749}, {2.32973, -1.13296},
 {-9.28741, 2.6097}, {-3.3001, 1.31403}, {3.28091, -6.56867}, {-4.45742, -3.50691},
 {-5.22019, -7.31909}, {8.72281, 2.20909}, {-1.75738, 8.50959}, {4.91354, 7.45721},
 {5.48025, -1.42342}, {9.46615, 4.72104}, {7.33425, -1.08151}, {5.55214, 2.43006},
 {8.78305, -8.43983}, {-1.63389, -5.64446}, {-6.33726, 6.14862},
 {-5.05503, 7.83732}, {9.89825, -6.70792}, {-1.25488, 5.78948}, {-6.64355, -4.31986},
 {5.54391, 5.09748}, {4.97521, -7.33067}, {-5.96476, 0.273494}, {4.95501, -9.22198},
 {-4.24909, -6.01423}, {1.07997, -0.97207}, {-0.425487, 1.10524}, {4.61603, -1.08979},
 {-4.19288, -4.14596}, {0.206045, -3.44104}, {-5.51746, -3.39021},
 {4.92202, -8.64063}, {-2.70173, 2.66111}, {7.56427, 3.89614}, {4.80096, -7.25148},
 {-9.66792, 2.19267}, {-8.06609, 2.25114}, {5.52372, 9.20761}, {-7.88423, 4.23404},
 {-6.21864, 6.96702}, {5.57343, 7.28515}, {9.36244, -4.81503}, {6.72125, -0.399189}}
```

```
{13.5889, 9.8086, -10.5601, -9.72236, 0.188278, 0.989546, 7.36917, 1.19678, -6.67771,
 -1.98607, -3.28776, -7.96432, -12.5393, 10.9319, 6.75221, 12.3708, 4.05682,
 14.1872, 6.25274, 7.9822, 0.343213, -7.27836, -0.188635, 2.78229, 3.19033, 4.53461,
 -10.9634, 10.6414, -2.35546, -5.69127, -4.26698, -10.2633, 0.107901, 0.679749,
 3.52624, -8.33884, -3.235, -8.90766, -3.7186, -0.040626, 11.4604, -2.45052,
 -7.47525, -5.81495, 14.7313, -3.65019, 0.748378, 12.8586, 4.54741, 6.32206}
```

Packages

Packages

- ***Mathematica*** **Packages (*Mathematica* Tutorial)**
- **Package Development (*Mathematica* Guide)**
- **A sample package**

Sample.m

```
<< Sample`
```

```
Context[AddTwo]
```

```
Sample`
```

```
AddTwo[a, b]
```

```
AddTwo[a, b]
```

```
Remove[Sample`AddTwo]
```

```
Remove[Global`AddTwo]
```

Communication with External Programs

- **Introduction to *MathLink* (*Mathematica* Tutorial)**
- ***MathLink*** **C API Developer Guide (Windows)**
- ***MathLink*** **API (*Mathematica* Guide)**
- **Example (addtwo.exe)**
- **addtwo.c**

```
#include "mathlink.h"
extern int addtwo( int i, int j);
int addtwo( int i, int j)
{
        return i+j;
}
```

- **addtwo.tm**

```
int addtwo P(( int, int));
:Begin:
:Function:       addtwo
:Pattern:        AddTwo[i_Integer, j_Integer]
:Arguments:      { i, j }
:ArgumentTypes:  { Integer, Integer }
:ReturnType:     Integer
:End:
:Evaluate: AddTwo::usage = "AddTwo[x, y] gives the sum of two machine integers x and y."
```

- **Compile and link**

```
SET CL=/nologo /c /DWIN32 /D_WINDOWS /W3 /O2 /DNDEBUG /I\Include
SET LINK=/NOLOGO /SUBSYSTEM:windows /INCREMENTAL:no kernel32.lib
user32.lib gdi32.lib /LIBPATH:\Lib
MPREP addtwo.tm -o addtwotm.c
CL addtwo.c addtwotm.c
LINK addtwo.obj addtwotm.obj ml32i2m.lib /OUT:addtwo.exe
```

- **Install and run**

Install

```
mlelink = Install["I:/Mathematica/MathLink/MathLinkExamples/addtwo/addtwo.exe"];
```

The two arguments must be integers.

```
AddTwo[1, 2]
```

```
3
```

```
AddTwo[1.0, 2.0]
```

```
AddTwo[1., 2.]
```

Uninstall.

```
Uninstall[mlelink]
```

```
"I:\Mathematica\MathLink\MathLinkExamples\addtwo\addtwo.exe"
```

Example (Modeling of Optical Fiber Modes)

Initialization

• Open the *MathLink* connection.

– Windows

```
mlelink = Install["mle.exe"];
```

– Unix

```
mlelink = Install[LinkConnect["aaa"]];
```

– Uninstall

```
Uninstall[mlelink]
```

```
mle.exe
```

• Set the default directory.

```
SetDirectory["D:/Temp"]
```

```
D:\Temp
```

```
Directory[]
```

```
D:\Temp
```

• Turn off Sellmeier formula by default.

```
DspnSetSellmeier[False];
```

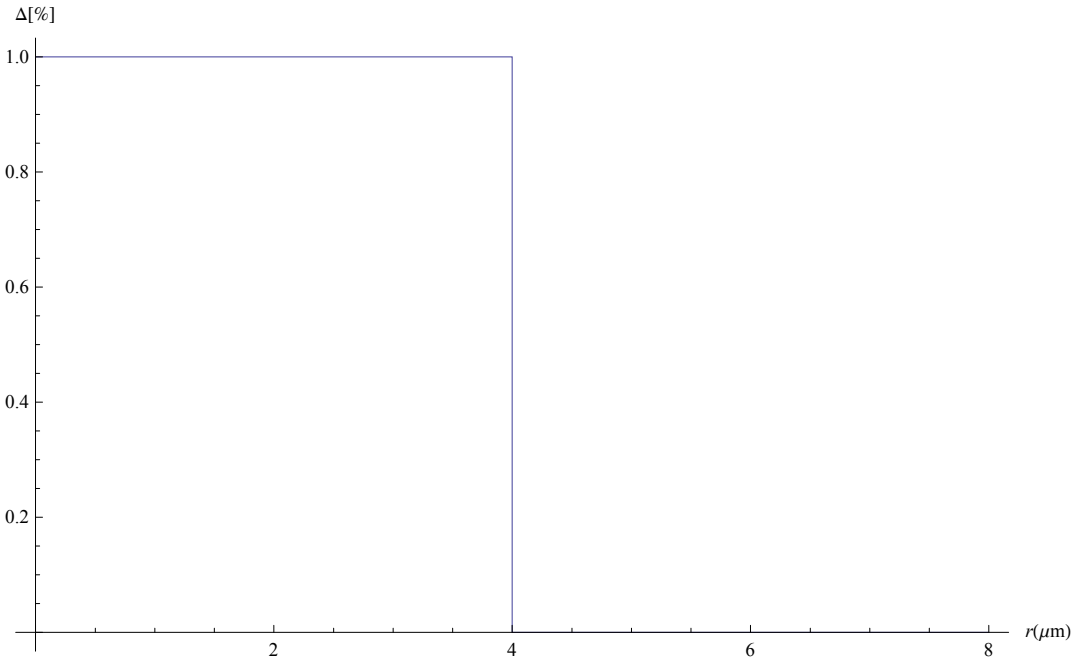
Refractive Index Profiles

• Step index

The data format for the step index profile is
{Step, n_{core}, {$a_{\text{core-clad}}$, n_{clad}}}
or
{Step, Δ_{core}, {$a_{\text{core-clad}}$, Δ_{clad}}}.
Δ is given in percent (%).

```
profile = {Step, 1, {4, 0}}
DspnPlotIndex[1.55, profile, Scale → Delta,
 AxesLabel → {r[μm], "Δ[%]"}, ImageSize → 500]
```

```
{Step, 1, {4, 0}}
```

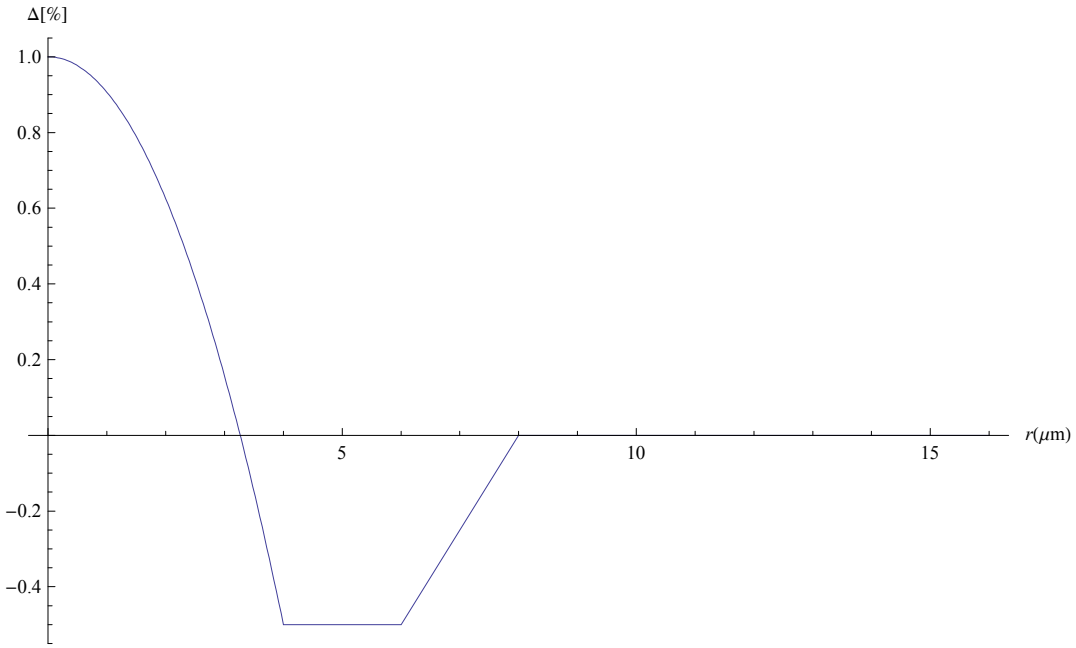


• α-index

The data format for the α index profile is
{Alpha, α, {{0, n_{core}}, {a_{core}, n_1}, {r_2, n_2}, ... , {r_N, n_N}, {$a_{\text{core-clad}}$, n_{clad}}}
or
{Alpha, α, {{0, Δ_{core}}, {a_{core}, Δ_1}, {r_2, Δ_2}, ... , {r_N, Δ_N}, {$a_{\text{core-clad}}$, Δ_{clad}}}.

```
profile = {Alpha, 2, {{0, 1}, {4, -0.5}, {6, -0.5}, {8, 0}}}
DspnPlotIndex[1.55, profile, Scale → Delta,
 AxesLabel → {r[μm], "Δ[%]"}, ImageSize → 500]
```

```
{Alpha, 2, {{0, 1}, {4, -0.5}, {6, -0.5}, {8, 0}}}
```

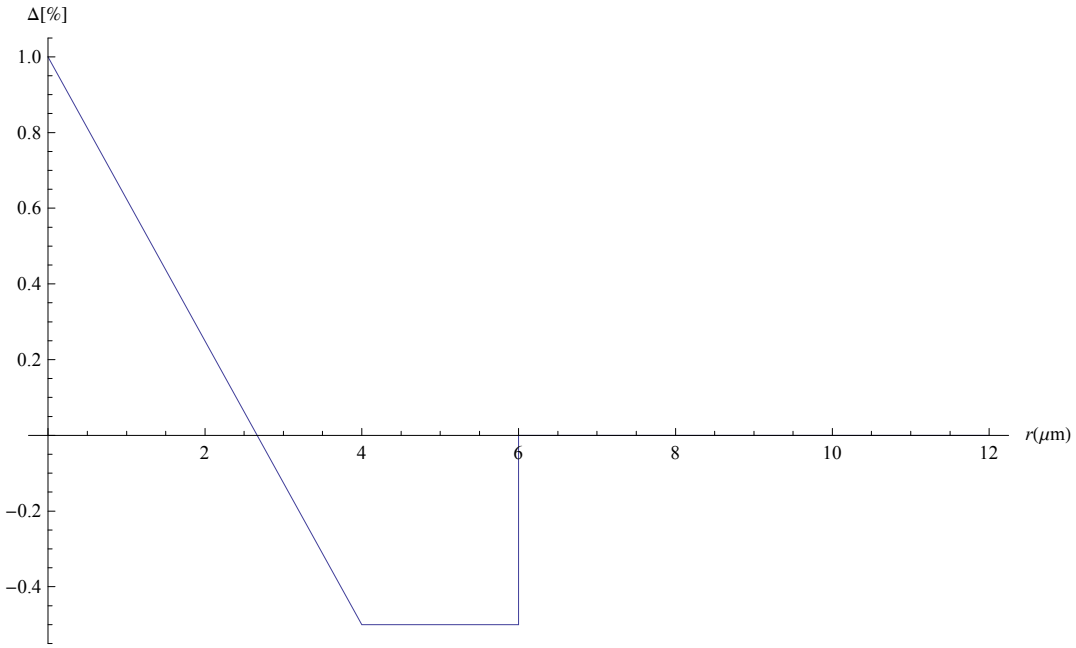


• General index

The data format for the general index profile is
{General, $\{0, n_{\text{core}}\}$, $\{r_1, n_1\}$, $\{r_2, n_2\}$, ... , $\{r_N, n_N\}$, $\{a_{\text{core-clad}}, n_{\text{clad}}\}$}
or
{General, $\{0, \Delta_{\text{core}}\}$, $\{r_1, \Delta_1\}$, $\{r_2, \Delta_2\}$, ... , $\{r_N, \Delta_N\}$, $\{a_{\text{core-clad}}, \Delta_{\text{clad}}\}$}.

```
profile = {General, {{0, 1}, {4, -0.5}, {6, -0.5}, {6, 0}}}
DspnPlotIndex[1.55, profile, Scale → Delta,
 AxesLabel → {r[μm], "Δ[%]"}, ImageSize → 500]
```

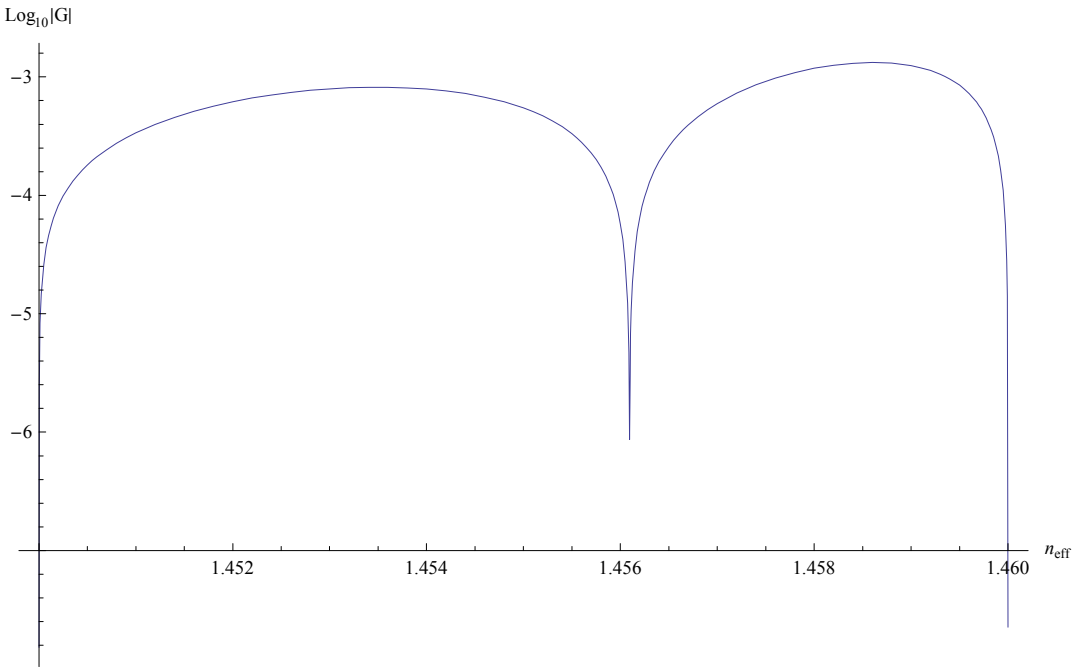
```
{General, {{0, 1}, {4, -0.5}, {6, -0.5}, {6, 0}}}
```



■ **Characteristic Equation**

• Plot the characteristic equation.

```
profile = {Step, 1.46, {4, 1.45}};
DspnPlotCharEquation[1, 1.55, profile,
 AxesLabel → {"neff", "Log10|G|"}, ImageSize → 500]
```



■ **Effective Index, Group Index, and Dispersion**

Effective Index, Group Index, and Dispersion

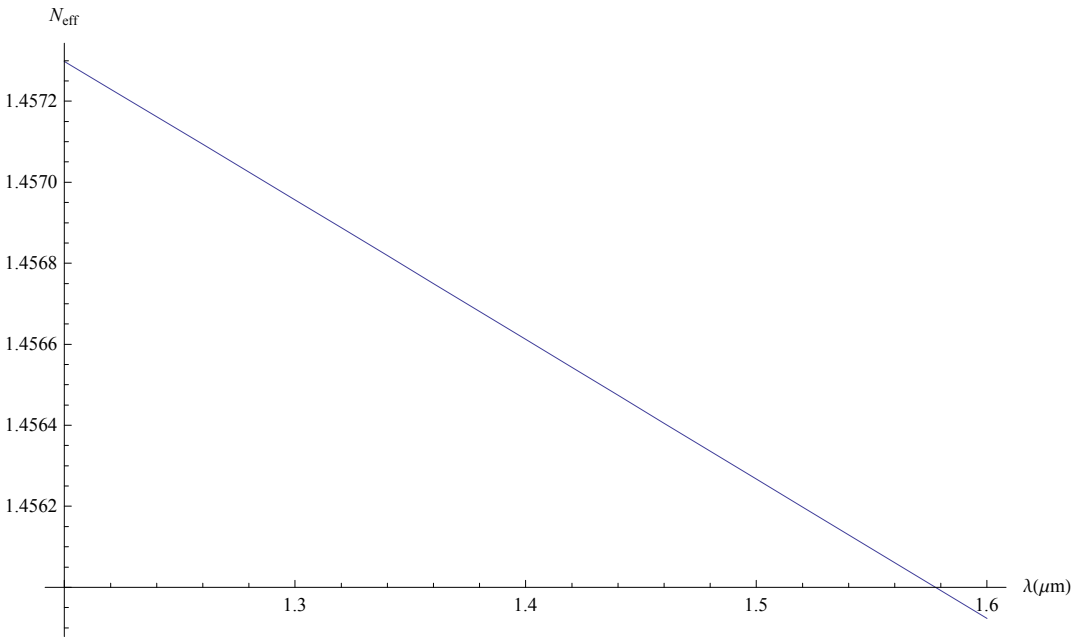
• Calculate the effective indexes, group indexes and dispersion of the guided modes.

```
profile = {Step, 1.46, {4, 1.45}};
sols = DspnNeffNgroupDspn[1, {1.2, 1.6, 0.02}, profile, NumModes → 1]
```

```
{Mode → {HE1,1}, Lambda → {1.2, 1.22, 1.24, 1.26, 1.28, 1.3, 1.32, 1.34,
   1.36, 1.38, 1.4, 1.42, 1.44, 1.46, 1.48, 1.5, 1.52, 1.54, 1.56, 1.58, 1.6},
 Neff → {{1.4573, 1.45723, 1.45716, 1.45709, 1.45702, 1.45696, 1.45689,
    1.45682, 1.45675, 1.45668, 1.45661, 1.45654, 1.45647, 1.45641,
    1.45634, 1.45627, 1.4562, 1.45613, 1.45606, 1.45599, 1.45592}},
 Ngroup → {{1.46137, 1.46138, 1.46139, 1.4614, 1.46141, 1.46142, 1.46143,
    1.46143, 1.46144, 1.46144, 1.46144, 1.46144, 1.46144, 1.46144,
    1.46144, 1.46143, 1.46143, 1.46142, 1.46142, 1.46141, 1.4614}},
 Dspn → {{2.10049, 1.93054, 1.75769, 1.58207, 1.4038, 1.22301, 1.03981, 0.854336,
    0.666699, 0.477018, 0.285407, 0.0919793, -0.103155, -0.299888, -0.498114,
    -0.697731, -0.898639, -1.10074, -1.30395, -1.50817, -1.71333}}}
```

• Plot the effective index N_{eff}

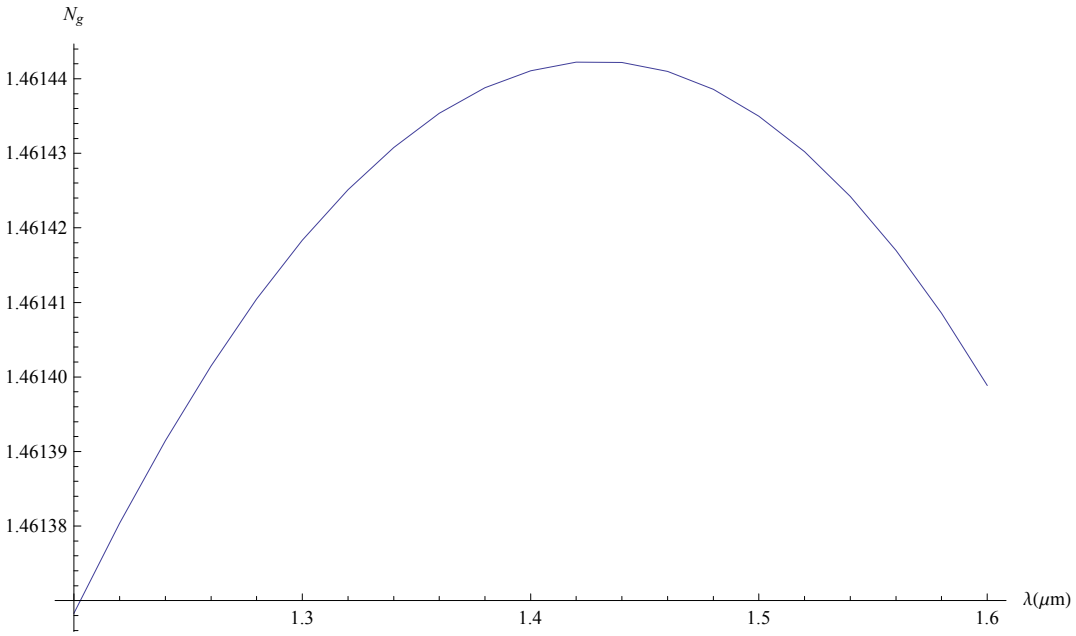
```
ListPlot[Transpose[{Lambda /. sols, (Neff /. sols)[[1]]}],
 Joined → True, AxesLabel → {λ[μm], Neff}, ImageSize → 500]
```



• Plot the group index N_g

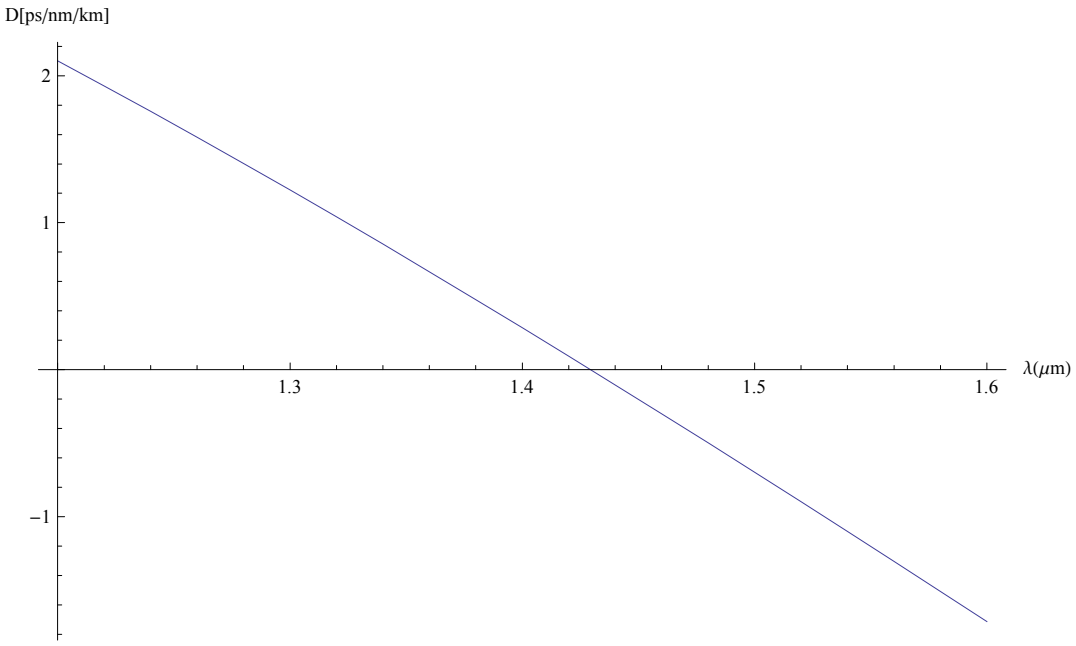
■

```
ListPlot[Transpose[{Lambda /. sols, (Ngroup /. sols)[[1]]}],
 Joined → True, AxesLabel → {λ[μm], Ng}, ImageSize → 500]
```



• Plot the dispersion *D*

```
ListPlot[Transpose[{Lambda /. sols, (Dspn /. sols)[[1]]}],
 Joined → True, AxesLabel → {λ[μm], "D[ps/nm/km]"}, ImageSize → 500]
```



■ **Poynting Vector and Field Profile**

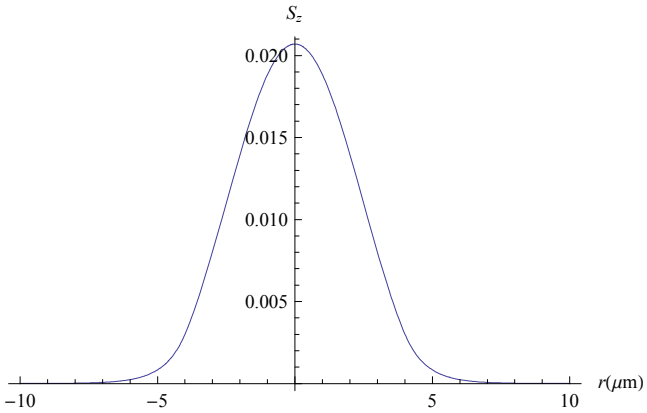
• Calculate the Poynting vector.

```
profile = {Step, 1.46, {4, 1.45}};
sols = DspnNeff[1, 1.55, profile]
SetCoordinates[Cartesian[x, y, z]];
ssol = DspnPoyntingVector[1, 1.55, 1, profile, Phase → π / 2, MaxRadius → 20];
```

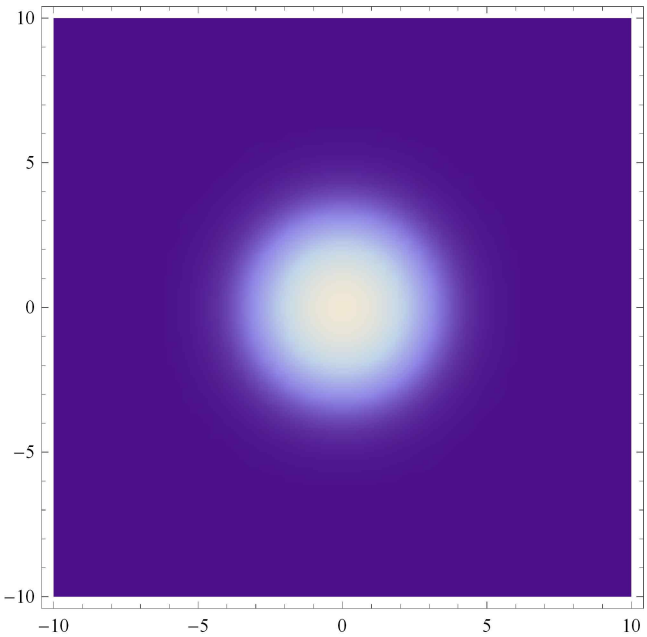
```
{Mode → {HE1,1}, Lambda → 1.55, Neff → {1.4561}}
```

• Plot the Poynting vector

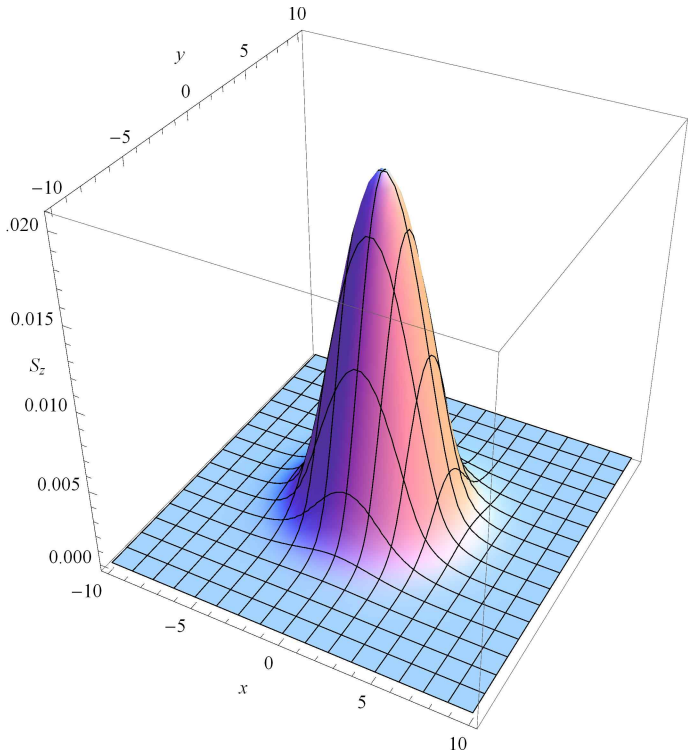
```
Plot[(PoyntingVector[[3]] /. ssol)[x, 0], {x, -10, 10}, AxesLabel → {r[μm], Sz}]
```




```
DensityPlot[(PoyntingVector[[3]] /. ssol)[x, y],
 {x, -10, 10}, {y, -10, 10}, PlotPoints → 50, PlotRange → All]
```



```
Plot3D[(PoyntingVector[[3]] /. ssol)[x, y], {x, -10, 10}, {y, -10, 10},
 AxesLabel → {x, y, Sz}, BoxRatios → {1, 1, 1}, PlotPoints → 50, PlotRange → All]
```



■ Example (Hardware Control via GPIB)

■ Initialization

• Open the *MathLink* connection.

```
mlelink = Install["mle.exe"];
```

```
mlelink = Install[LinkConnect["aaa"]];
```

```
Uninstall[mlelink]
```

```
mle.exe
```

• Initialization of the GPIB interface for data acquisition and control.

```
GPIBInitialize[{"GPIB0", 0, IsBoard → True},
 {"tds360", 1, TimeOut → 11}, {"hp3245a", 9, TimeOut → 11}]
```

```
board = "GPIB0", address = 0, eos = 10, timeout = 10
device = "tds360", address = 1, board = "GPIB0", eos = 10, timeout = 11
device = "hp3245a", address = 9, board = "GPIB0", eos = 10, timeout = 11
```

```
GPIBIfClear["GPIB0"]
```

```
Interface cleared: board "GPIB0".
```

• Set the default directory.

```
SetDirectory["D:/Temp"]
```

```
D:\Temp
```

```
Directory[]
```

```
D:\Temp
```

■ **Communication Examples**

• Identfication of the equipment

```
GPIBClear["tds360"];
GPIBWrite["tds360", "*IDN?"];
GPIBRead["tds360"]
```

```
TEKTRONIX,TDS 360,0,CF:91.1CT FV:v1.09
```

• Retrieve the data from the equipment and store the data to a file.

```
GPIBWrite["tds360", "CURVe?"];
GPIBReadFile["tds360", "data.txt", TimeOut → 11, Delete → True]
```

```
Read to file: device "tds360" to file "D:\Temp\data.txt".
```

• Switch the data source to Channel 1.

```
GPIBWrite["tds360", "DATa:SOUrce CH1"];
```

• Verify the data source.

```
GPIBWrite["tds360", "DATa:SOUrce?"];
GPIBRead["tds360"]
```

```
:DAT:SOU CH1
```

■ **Square Pulses and the Fourier Spectrum**

• Set the amplitude and frequency of the signal source.

```
GPIBWrite["hp3245a", "RST"];
GPIBWrite["hp3245a", "APPLY SQV 1.0"];
GPIBWrite["hp3245a", "FREQ 1E3"];
```

• Read the data from the oscilloscope and the store the data in the variable "data".

```
GPIBWrite["tds360", "CURVe?"];
data = GPIBRead["tds360", TimeOut → 11];
```

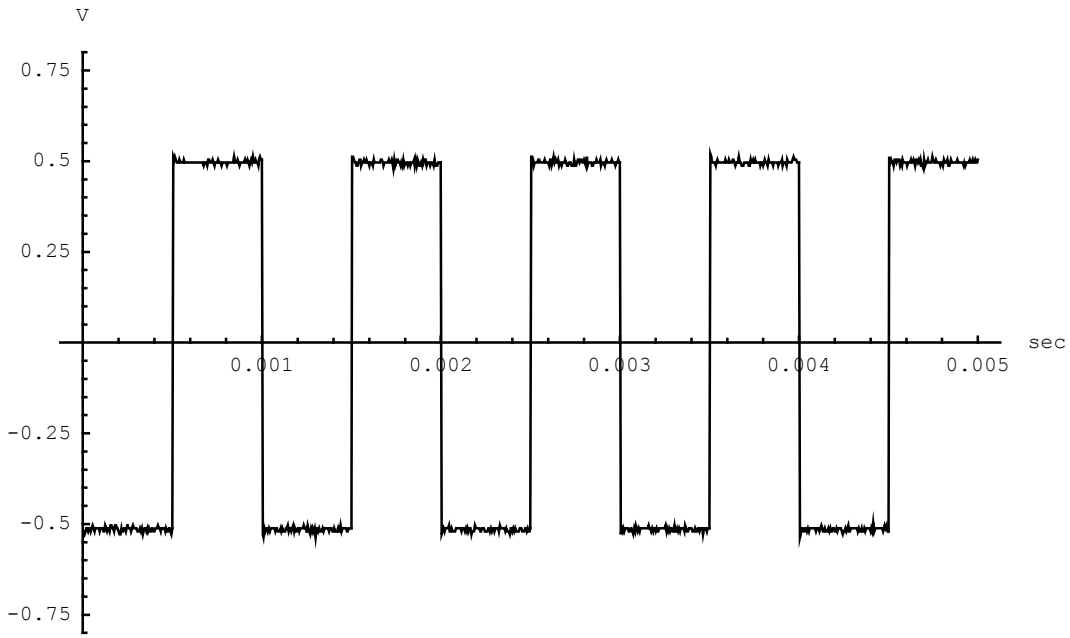
• Parse the data and put the t and y data in the variable " pdata". The format of "pdata" is $\{\{t_1, y_1\}, \{t_2, y_2\}, \ldots\}$. The time t_i is in the unit of second and the data y_i is in the unit of volt.

```
GPIBClear["tds360"];
GPIBWrite["tds360", "WFMPre:CH1:XINcr?"];
xinc = ToExpression[
   StringReplace[Part[StringSplit[GPIBRead["tds360"], " "], 2], "E" → "*^"]];
GPIBWrite["tds360", "WFMPre:CH1:YMUlt?"];
ymu = ToExpression[StringReplace[
    Part[StringSplit[GPIBRead["tds360", TimeOut → 12], " "], 2], "E" → "*^"]];
adata = StringDrop[data, 6];
pdata = ToExpression /@ StringSplit[adata, ","];
pdata = Table[{xinc * i, ymu * pdata[[i]]}, {i, Length[pdata]}];
Length[pdata]
```

```
1000
```

• Plot the data.

```
ListPlot[pdata, Joined → True, ImageSize → 500,
 PlotRange → {-0.8, 0.8}, AxesLabel → {"sec", "V"}]
```




```
- Graphics -
```

• Fourier transform. First separate the time and signal data. Put the time data in the variable "t" and the signal data in the variable "y".

```
{t, y} = Transpose[pdata];
```

The frequency data

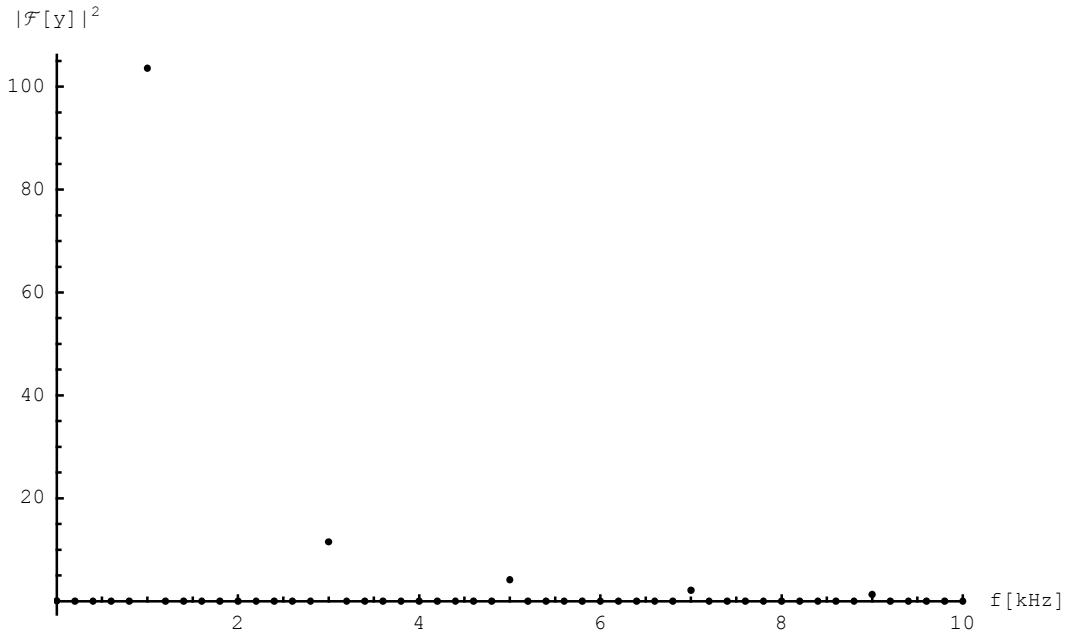
```
f = 1/(Length[t] (t[[2]] - t[[1]])) (Range[Length[t]] - 1);
```

The Fourier transform of the signal data

```
fy = Fourier[y];
```

Plot the spectrum data.

```
ListPlot[Transpose@{f * 0.001, (Abs /@ fy)^2}, PlotRange → {{0, 10}, All},
 ImageSize → 500, AxesLabel → {"f[kHz]", "|ℱ[y]|^2"}]
```



- Graphics -

• Clear the data.

```
f =.; t =.;
```

■ **Sinusoidal Waves and Curve Fitting**

• Set the amplitude and frequency of the signal source.

```
GPIBWrite["hp3245a", "RST"];
GPIBWrite["hp3245a", "APPLY ACV 1.0"];
GPIBWrite["hp3245a", "FREQ 1E3"];
```

• Read the data from the oscilloscope and the store the data in the variable "data".

```
GPIBWrite["tds360", "CURVe?"];
data = GPIBRead["tds360", TimeOut → 11];
```

• Parse the data and put the t and y data in the variable " pdata". The format of "pdata" is $\{\{t_1, y_1\}, \{t_2, y_2\}, \ldots\}$. The time t_i is in the unit of second and the data y_i is in the unit of volt.

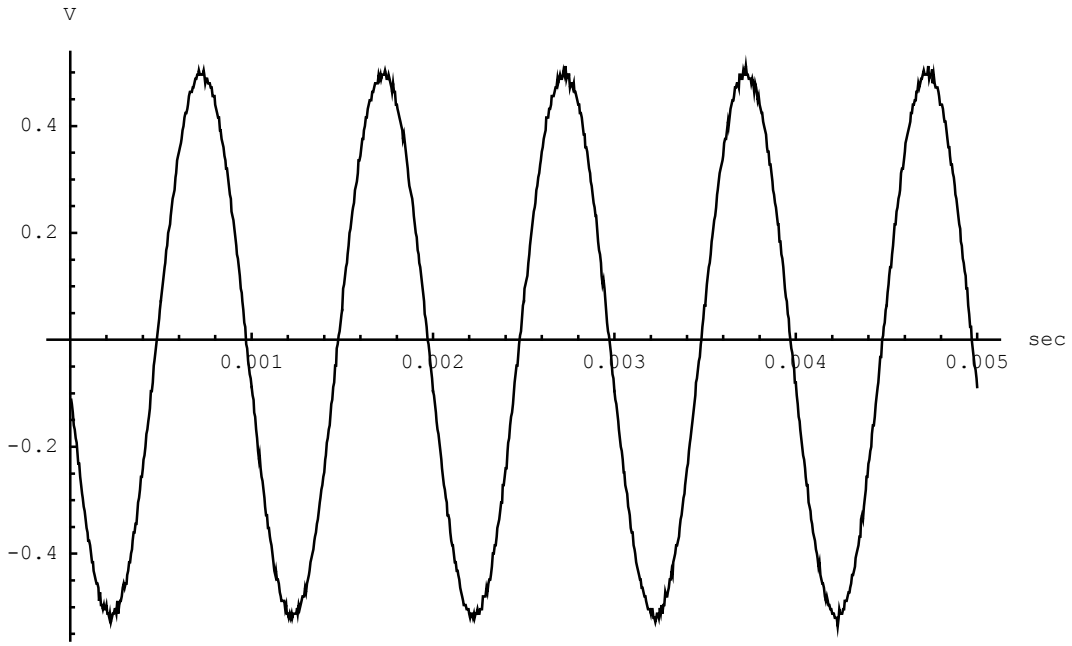
• Parse the data and put the t and y data in the variable " pdata". The format of "pdata" is $\{\{t_1, y_1\}, \{t_2, y_2\}, \ldots\}$. The time t_i is in the unit of second and the data y_i is in the unit of volt.

```
GPIBClear["tds360"];
GPIBWrite["tds360", "WFMPre:CH1:XINcr?"];
xinc = ToExpression[
   StringReplace[Part[StringSplit[GPIBRead["tds360"], " "], 2], "E" → "*^"]];
GPIBWrite["tds360", "WFMPre:CH1:YMUlt?"];
ymu = ToExpression[StringReplace[
    Part[StringSplit[GPIBRead["tds360", TimeOut → 12], " "], 2], "E" → "*^"]];
adata = StringDrop[data, 6];
pdata = ToExpression /@ StringSplit[adata, ","];
pdata = Table[{xinc * i, ymu * pdata[[i]]}, {i, Length[pdata]}];
Length[pdata]
```

```
1000
```

• Plot the data.

```
pl1 = ListPlot[pdata, Joined → True, ImageSize → 500, AxesLabel → {"sec", "V"}]
```




```
- Graphics -
```

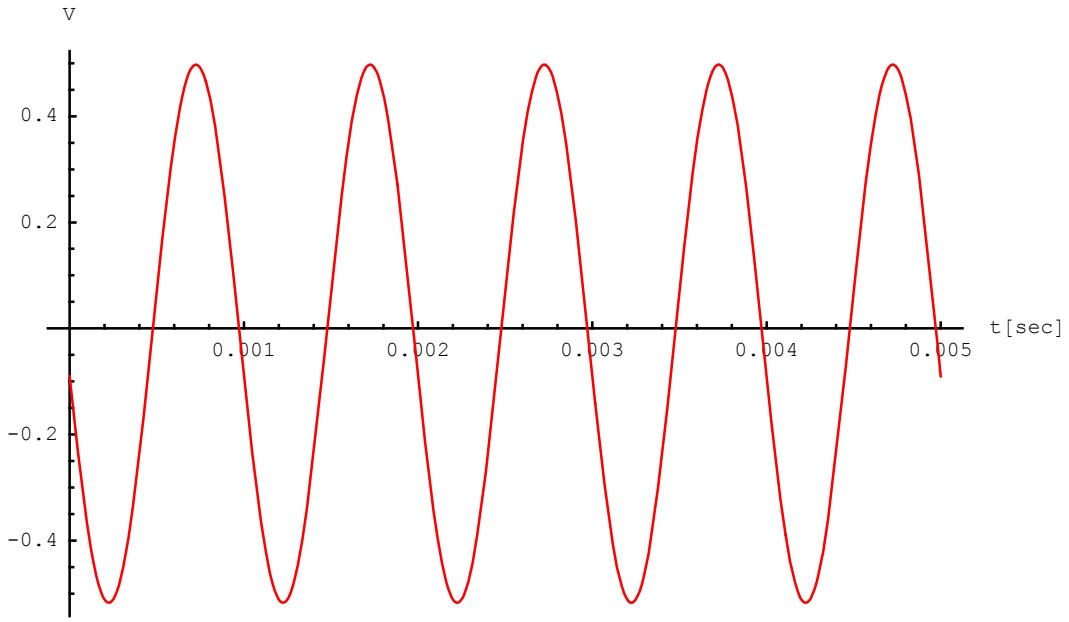
• Curve fitting. Fit the data to the sine curve and find the fitting parameters.

```
f = 1000;
fitparams = FindFit[pdata, A + B Sin[2 π f t + ϕ], {A, B, ϕ}, t]
```

```
{A → -0.009912, B → -0.507288, ϕ → 0.159094}
```

• Plot the sine curve using the fitting parameters.

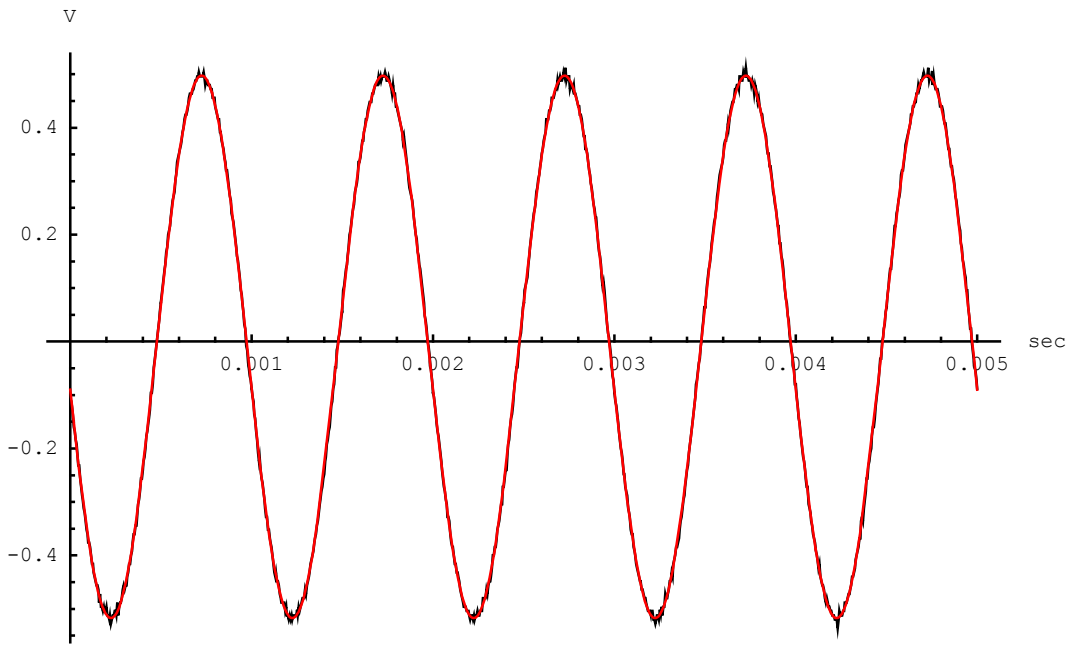
```
pl2 = Plot[Evaluate[A + B Sin[2 π f t + ϕ] /. fitparams], {t, 0, 0.005},
  ImageSize → 500, PlotStyle → {RGBColor[1, 0, 0]}, AxesLabel → {"t[sec]", "V"}]
```



- Graphics -

• Comparison of the original data and the fitting curve.

```
Show[{pl1, pl2}]
```



- Graphics -

If you want to know more about symbolic computing,

please visit:

http://ontl.gist.ac.kr
→ Lectures → Symbolic Computing Methods and Applications