A Brief Introducion to MAPLE™

1 Introduction

MAPLE™ is a *symbolic maniplation package* released under commercial license by MAPLESOFT. In its most basic form, as in other symbolic manipulation softwares, it can perform a large number of mathematical operations on symbols representing variables and on strings consisting of several symbols. This is in contrast to common numerical computation packages which perform mathematical operations on numbers. The output of MAPLE is in the form of symbols and strings which can be evaluated by assigning numbers to the symbols and strings. In its current sophisticated from, MAPLE can integrate seamlessly numerical and symbolic calculations, text and math, graphics and images, and several other kinds of information. For more details, visit Maplesoft Official Website. In the context of robotics and the material contained in the Modules, MAPLE has been used extensively to obtain and solve the algebraic equations arising in kinematic analysis of serial and parallel manipulators. It has been used extensively to obtain dynamic equations of motion of manipulators and for dynamic analysis of manipulators. The capability of symbolic computations allows one to develope "error free" equations in kinematics and dynamics of manipulators, which in turn is solved to obtain numerical results when required. This tutorial is intended as a brief introduction to using MAPLE, specifically in the context of robotics, although it's applications are much more wider.

2 Using Maple

At the time of writing this tutorial, the running version was Maple 14. The company offers subsidized versions of the software to students for academic/non-commercial usage. Please check the official site for pricing and other details. Once a copy is obtained, install the software following the instructions in their installation guide.

Once installed, the software may be launched by

1. **Windows (All Versions):** *Start Menu→Programs→Maplesoft→Maple (version number) or just launching the short-cut(if it exists)on the desktop.*

2. **Linux/Unix:** *Menu→Maple (version number) or just launching the shortcut(if it exists) on the desktop.*

The GUI comes up as shown in the screenshot below.

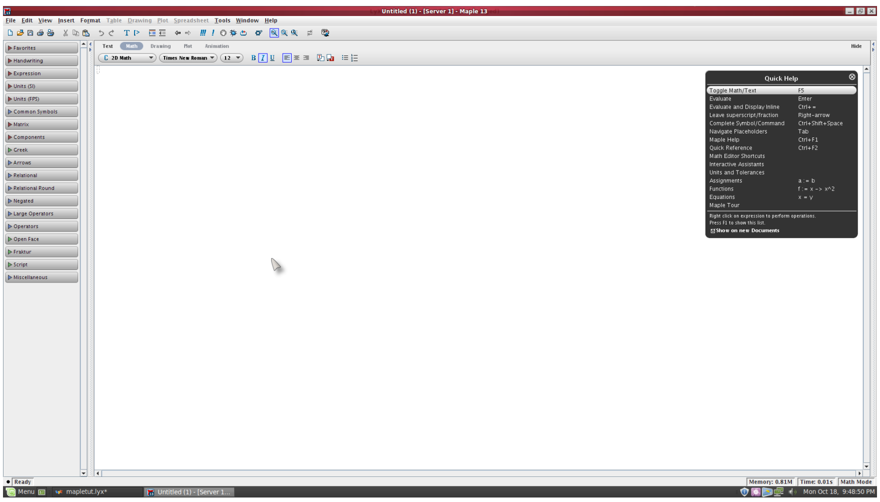

Figure 1: MAPLE GUI

2.1 Different modes/environments available in MAPLE

MAPLE has two interfaces for working namely the notebook mode & the worksheet mode.

1. **Worksheet Mode:** This is the 'conventional' command line interface of MAPLE, included in the package since it's inception. This interface allows the users to enter all the commands using a keyboard alone, and calls for minimal use of menus & GUI.

(a) Traditional MAPLE problem-solving environment

(b) Enter problems at a prompt ([>)

(c) Math entered and displayed in 2-D or 1-D

(d) Solve math problems with right-click menu on output

2. **Document Mode:** This is the 'modern' & highly interactive interface of MAPLE & has been recently incorporated into the package. This interface allows the users to work with MAPLE just like one works on a paper by employing the highly dynamic & interactive GUI environment. User need not memorize complicated commands and only minimal commands need to be remembered. Other commands may be added from the menu driven interface. As MAPLE quotes

   (a) Quick problem-solving and free-form, rich content composition

   (b) No prompt ([>) displayed

   (c) Math is entered and displayed in 2-D

   (d) Solve math problems with right-click menu on input and output

As far as this tutorial is concerned, we'll be dealing with the minimal and traditional worksheet mode of MAPLE. This is because it is the more widely accepted method as of today and, in addition, the notebook mode has only been recently introduced feature. It is present only in the latest versions of the software.

To start a file in MAPLE in worksheet mode, select ***File→New→Worksheet Mode.***

2.2 Basic commands

Once a new worksheet is opened, a command prompt appears as shown below.

[>

This is the MAPLE command prompt and we will have to execute commands and computations at this prompt. Any MAPLE code is a series of expressions. Each expression must be explicitly terminated with a semicolon ';' (using a colon ':' instead supresses the output). If a semicolon or colon is not present at the end of a command, MAPLE will interpret the next command line as a continuation of the previous command and the first command won't be executed immediately. Expressions will only be evaluated or simplified to a certain point automatically; further simplifications must be explicitly asked for.
Tip: Once you've done something in a program step (like redefine a variable), it stays done unless you change it again. Even if you click on an earlier step and re-execute it, the effect of the change remains. If a program does not seem to be working, before trying anything else re-execute all steps from the beginning by clicking on the *Edit→Execute Worksheet*. (You can also select a set of commands and re-execute them by selecting *Execute→Selection*.)

Basics

Variables like `x`, `f`, `a`, `bar`, etc. are simply symbols by default. They can also be assigned values with the `':='` operator. Similar to functional programming languages like *Scheme* or *Lisp*, variables have no type, so they can refer to any expression (symbols, numbers, matrices, procedures...). However, every expression in MAPLE does have a type – can be checked by typing out '`whattype`' at the command prompt.

By default, all the simplifications MAPLE does are *exact and symbolic*. Rational, complex, and algebraic numbers are all supported. To find an approximation, use command `'evalf'`. To evaluate an expression, substituting a specific value for one variable, use command `'eval`'.

Sometimes a group of functions is grouped into a package or module. To access these, you can use the long form of the command, such as `LinearAlgebra[CharacteristicPolynomial]`, or use the "with" command, as in `with(LinearAlgebra)`, to lift those functions to the top namespace.

Vectors, Matrices & Linear Algebra

Matrices have gone through a few historical eras in MAPLE. We will be using the current preferred representation, from the `LinearAlgebra` package. Note that this is not the same as the old `linalg` package. Our matrices and vectors will be of type `Matrix` and `Vector` resp., not `matrix` and `vector`. This important point is to be noted. Additional information on differences between the two packages can be obtained from the following URL's.

- A Short Extract from Dr. Francis Wright's 'Computing in MAPLE'
- Comparing the `linalg` and `LinearAlgebra` packages

When one types "with(LinearAlgebra)" a large number of routines are mported into the namespace. Most of the routines one needs are present – the most useful ones are listed at the end of this section for reference.

To create a Matrix or a Vector, one can use the functions `Matrix` and `Vector`, which give many options for instantiating the structures. For small, fixed Matrices and Vectors, one can use the shortcuts `<...>`, with `,` separating the entries of a column vector, and `|` separating the entries of a row vector. A Matrix is just a row vector of column vectors, or a column vector of row vectors. For example, the code `<<1,4>|<2,5>| <3,6>>` produces the matrix

$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix}$$

To access the entries of a Matrix or Vector, use the subscript brackets `[]` just as with lists or sets. Of course, Matrices will need two indices. Note that the entries of a Matrix by default are indexed in row-major order; that is, the first index selects the row, and the second index selects the column within that row. And again the indices start at 1. The dimensions can be obtained with the `Dimensions` command.

The functions for arithmetic on Matrices and Vectors are all in the `LinearAlgebra` package – type `?LinearAlgebra` or `with(LinearAlgebra);` to see a listing of them. Note that this includes basic arithmetic such as `Multiply` and `Add`, as well as matrix invariant computations such as `Determinant` and `Rank`. Another computation that is useful is `LinearSolve`, which gives a finds a solution vector x to the system $Ax = b$, for a given matrix A and vector b.

Miscelleanous

For running big computations, one can prevent "bytes used" messages with `kernelopts(printbytes=false)`. For timing a program, use the `time()` command to get the total CPU time used so far in the execution.

Appendix of Commonly used Commands

Given below are some of the most commonly used commands in MAPLE. Type `?command` followed by the command name to get a synopsis of how they work.

- **Basic Arithmetic:** `+`, `-`, `*`, `/`, `^`, `min`, `max`, `abs`, `I`, `eval`, `subs`, `solve`, `expand`, `simplify`, `sum`, `product`, `numer` , `denom`, `normal`
- **Floating Point:** `evalf`, `Digits`, `sqrt`, `log`, `log[b]`, `floor`, `ceil`, `Pi`
- **Integers:** `igcd`, `igcdex`, `ilcm`, `iquo`, `irem`, `isprime`, `nextprime`, `ifactor`, `rand`, `factorial`, `binomial`
- **Modular Arithmetic:** `mod`, `mods`, `modp`, `&^`
- **Programming:** `proc`, `return`, `if`, `for`, `while`, `do`, `local`, `global`, `nargs`, `error`, `print`, `lprint`
- **Comparison:** `=`, `<`, `>`, `<=`, `>=`, `<>`
- **Data:** `:=`, ’’, `""`, ‘‘, `whattype`, `type`
- **Lists, Sets, Sequences:** `[]`, `{}`, `seq`, `nops`, `op`, `union`, `intersect`, `minus`, `subset`
- **Polynomials:** `degree`, `sort`, `coeffs`, `lcoeff`, `tcoeff`, `indets`, `collect`, `content`, `primpart`, `with(PolynomialTools)`
- **Polynomial Arithmetic:** `gcd`, `gcdex`, `lcm`, `+`, `-`, `*`, `^`, `quo`, `rem`, `randpoly`, `factor`, `factors`, `irreduc`, `sqrfree`, `roots`, `modp1`, `ConvertIn`, `ConvertOut`
- **Inert Commands:** `Gcd`, `Gcdex`, `Lcm`, `Eval`, `Quo`, `Rem`, `Factor`, `Factors`, `Roots`, `...`
- **Linear Algebra:** `<<...>>`, `with(LinearAlgebra)`, `Matrix`, `Vector`, `Dimension`, `RowDimension`, `ColumnDimension`, `Determinant`, `Add`, `Multiply`, `MatrixInverse`, `RandomMatrix`, `LinearAlgebra[Modular]`, `Modular:-Copy`, `Modular:-Mod`, `Modular:-Create`
- **Miscellaneous:** `kernelopts(printbytes=false)`, `time`, `showtime`, `quit`, `read`, `save`, `unwith`, `restart`

List of Keyboard Shortcuts

The following are the key bindings/shortcuts useful while working in the MAPLE worksheet interface.

Sl. No.	Keyboard Shortcut	Function
1	Ctrl + B	Cursor Left
2	Ctrl + F	Cursor Right
3	Ctrl + A	Move to the Beginning of the Line
4	Ctrl + E	Move to the End of the Line
5	Ctrl + W	Move One Word Right
6	Ctrl + Y	Move One Word Left
7	Ctrl +]	Move to Matching Parenthesis, Brace or Square Bracket
8	Ctrl + D	Delete (to Right of Cursor)
9	Ctrl + H	Backspace (to Left of Cursor)

10	Ctrl + X or Ctrl + G	Clear the Line
11	Ctrl + K	Clear to the End of Line
12	Ctrl + U	Undo Changes to the Line
13	Ctrl + P	Previous Command From the History
14	Ctrl + N	Next Command From the History
15	Ctrl + R	Find Matching Command From the History
16	Ctrl + Space or Tab	Command Completion
17	Ctrl + T	Show Completion Matches
18	Ctrl + C	Interrupt the Currently Executing Command
19	Ctrl + _	Stop the Currently Executing Command in the Debugger
20	Ctrl + V	Toggle Insert or Overwrite Mode
21	Ctrl + L	Redraw the Current Prompt and any text entered

2.3 Tips for Troubleshooting in MAPLE

- Make sure there is a semicolon or a colon at the end of the command. Otherwise, MAPLE won't process the information. Remember that a colon will suppress the output so Maple will process the command but it won't show anything on the screen. With a semicolon, all relevant output should be displayed. Write the product 2x as 2*x.
- MAPLE follows the Orders of Operation (first do multiplication, then addition). If it seems to give wrong answers, then insert parentheses in the appropriate places.
- Don't forget about the command `evalf`. Without it, MAPLE may simplify the answer. The command `simplify` is also often useful.
- Remember that both `log` and `ln` use e as their base. `log` does not use base 10.
- To assign variable, expressions and functions, use the `:=` operator. A plain equals sign has a different meaning in MAPLE.
- If a function is to be used instead of an expression, remember to use functional notation consistently.
- Maple is case sensitive. "pi" is not the same thing as `Pi` – (`Pi` is the actual constant 3.14159...).
- When a saved worksheet in MAPLE is opened, run the `Format→Execute Worksheet` command from the menu, so that MAPLE "knows" everything that has been already worked on.
- If MAPLE is doing strange things to your variables, check through the rest of the worksheet and make sure that the same letter has not been used to name two different things. Because of this possibility, it's a good idea to save different problems in different files. Another way to solve this problem is to enter the command `restart` at the prompt. This will clear any values that have been previously assigned.

2.4 Sources for Help & References

MAPLE has an extensive and well-documented help system, which serves as the best and freely available readymade reference at your fingertips. This can be further supplemented by the many online introductory tutorials and books from various publishing houses, for those are interested. In addition, Maplesoft, the company behind Maple, offers free webinars, user case studies and application briefs in their site. Most importantly, they have posted more than 2100 preprogrammed MAPLE notebooks dealing with problems in highly diverse areas, which in turn may be accessed from the Maple Application Center and may be used as tutorials which can be used hands-on to demonstrate in MAPLE environment.

3 Examples

We have two examples in this tutorial. First is the classic planar 2R manipulator consisting of two-links and two revolute joints, and the second is the well-known one degree-of-freedom planar four-bar mechansm. In both the examples the dynamic equations of motion are derived in a symbolic form. It is assumed that the link parameters, and mass properties are known. Once the equations of motion are derived, they can be solved numerically in the MAPLE environment. The numerical solution is not discussed in this example.

3.1 2R Manipulator

Figure 2 shows the well-known planar 2R manipulator. The mass, length, location of centre of mass and I_{zz} component of inertia of the two links are denoted by $(m_i, l_i, r_i, I_i),\ i = 1, 2$. The generalised coordinates are $\theta_1(t)$ and $\theta_2(t)$ and the torques at the two joints are $\tau_1(t)$ and $\tau_2(t)$. The gravity acts along the negative $\hat{\mathbf{Y}}_0$ direction.

We present in detail the steps in MAPLE which give rise to the equations of motion for the planar 2R manipulator.

1. Open up the MAPLE environment in Worksheet mode and save the file with an appropriate name of your choice.

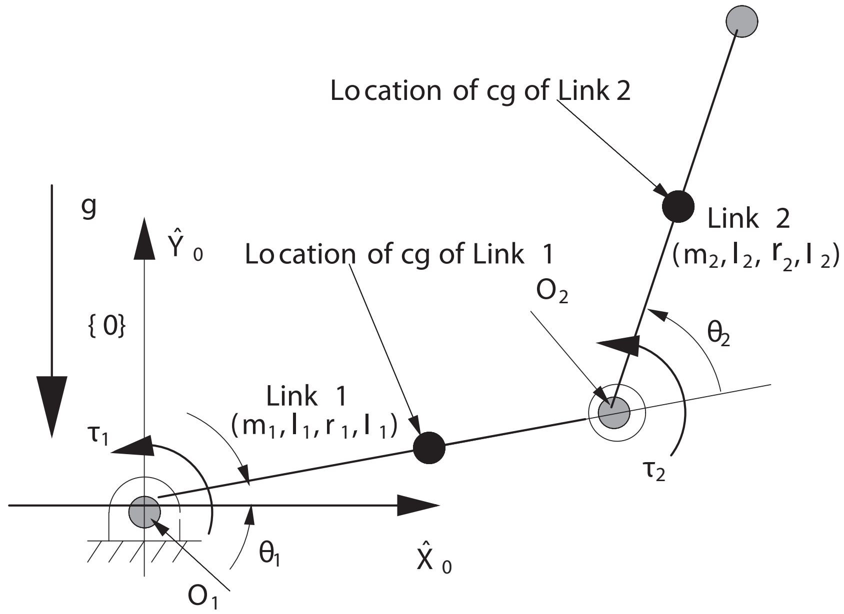

Figure 2: The planar 2R manipulator

2. We start by defining the kinetic energy of the 2R manipulator which is given by

$$KE = 0.5\, m_1 {r_1}^2 dtheta_1{}^2 + 0.5\, I_1\, dtheta_1{}^2 + 0.5\, I_2\, (dtheta_1 + dtheta_2)^2$$
$$+ 0.5\, m_2[{l_1}^2 dtheta_1{}^2 + {r_2}^2\, (dtheta_1 + dtheta_2)^2 + 2\, l_1 r_2 \cos(\theta_2)\, dtheta_1\, (dtheta_1 + dtheta_2)]$$

To define the above in MAPLE and assign it to variable KE, we give as follows.

```
[>KE:=.5*m[1]*(r[1]*dtheta[1])^2+.5*I[1]*dtheta[1]^2+.5*I[2]*(dtheta[1]+dtheta[2])^2
+.5*m[2]*[l[1]^2*dtheta[1]^2+r[2]^2*(dtheta[1]+dtheta[2])^2
+2*l[1]*r[2]*cos(theta[2])*dtheta[1]*(dtheta[1]+dtheta[2])]
```

Note: For producing a subscript, the text has to be enclosed in square brackets i.e. $dtheta_1$ is to be written as `dtheta[1]`. Also it may be noted that Greek alphabets like $\alpha, \beta, \gamma, \theta$ etc. can be used in MAPLE by giving typing out the names of the alphabet in english. For example, α may be spelt as `alpha`, likewise β as `beta`, γ as `gamma` etc.

Two notable and highly useful features introduced in MAPLE include what are called as *symbol selection* and *command completion*. These features act as an excellent aid for the programmer by helping in command completion, thereby removing the burden of remembering the command names from the programmer's shoulders. For example, if one does not know the complete input to be given for the symbol ϵ, all one needs to do is to enter leading characters of the command followed by [Esc] key (or [Ctrl][Shift][Space]), i.e. probably *eps* [Esc] in this case. The software provides an *in situ* list of the probable commands/ symbols starting with the letters *eps*, from which the user may select whatever he intended to type out.

3. Next, we define the Potential Energy of the manipulator which is given by

$$PE = m_1 g r_1 \sin(\theta_1) + m_2 g\, (l_1 \sin(\theta_1) + r_2 \sin(\theta_1 + \theta_2))$$

We define this in MAPLE as follows.

```
[>PE:=m[1]*g*r[1]*sin(theta[1])+m[2]*g*(l[1]*sin(theta[1])+r[2]*sin(theta[1]+theta[2]))
```

4. Next we obtain the Lagrangian of the system as the difference between kinetic and potential energies.

```
[>L:=KE-PE
```

5. Once the Lagrangian is derived, we can formulate the Euler-Lagrange(E-L) equations. The E-L equations are of the form

$$\frac{d}{dt}\left(\frac{\partial \mathcal{L}}{\partial \dot{q}_i}\right) - \frac{d\mathcal{L}}{dq_i} = \tau_i(t)\ for\ i = 1\ to\ \mathrm{dof}$$

As a part of deriving these equations, symbolic differentiation of the Lagrangian with respect to joint variables (q_i) and its derivative ($\dot{q}_i$) is required. The angular rates have been denoted as *dtheta1* and *dtheta2* in the expression for Lagrangian. MAPLE needs to be informed that these along with the variables θ_1 & θ_2 are variables of time. These variables have to be properly substituted with subexpressions which indicate them as explicit time variables. This is where `subs` come into picture. According to MAPLE, this command is used to substitute sub-expressions into an expression. The syntax is `subs(substituting expression, main expression)`. Therefore we define the substituting expression as

```
[>symb_time:={dtheta[1]=diff(theta[1](t),t), dtheta[2]=diff(theta[2](t),t),
theta[1]=theta[1](t), theta[2]=theta[2](t)}
```

6. Next, we evaluate the term inside the parentheses in the E-L equations, namely $\frac{\partial \mathcal{L}}{\partial \dot{q}_i}$, by the following command.

```
[>temp[1]:=collect(simplify(subs(symb_time, diff(L, dtheta[1]))),{m[1], m[2]})
```

It may be noted that the commands `collect` & `simplify` do just what their names indicate, namely collect terms simplify exressions at an elementary levelsuch as expanding square of sums, trigonometric expansions etc. More details on them may be obtained from MAPLE help.

The output for the expression will be

$$temp_1 = m_1 {r_1}^2 \frac{d}{dt}\theta_1(t) + [1.0\, {l_1}^2 \frac{d}{dt}\theta_1(t) + 1.0\, {r_2}^2 \left(\frac{d}{dt}\theta_1(t) + \frac{d}{dt}\theta_2(t)\right) + 1.0\, l_1 r_2 \cos(\theta_2(t)) \left(\frac{d}{dt}\theta_1(t) + \frac{d}{dt}\theta_2(t)\right) + 1.0\, l_1 r_2 \cos(\theta_2(t)) \frac{d}{dt}\theta_1(t)] m_2 + [0.0] + I_1 \frac{d}{dt}\theta_1(t) + I_2 \frac{d}{dt}\theta_1(t) + I_2 \frac{d}{dt}\theta_2(t)$$

Moving along similar lines, we derive the same term for the E-L equation with respect to θ_2 by the command

```
[>temp[2]:=collect(simplify(subs(symb_time, diff(L, dtheta[2]))), {m[1], m[2]})
```

Which in turn gives the output as,

$$temp_2 = [1.0\, {r_2}^2 \left(\frac{d}{dt}\theta_1(t) + \frac{d}{dt}\theta_2(t)\right) + 1.0\, l_1 r_2 \cos(\theta_2(t)) \frac{d}{dt}\theta_1(t)] m_2 + I_2 \frac{d}{dt}\theta_1(t) + I_2 \frac{d}{dt}\theta_2(t) + [0.0]$$

7. Next, we derive the final form of the first E-L Lagrange equation for the system by entering the following.

```
[>eq[1]:=collect(simplify(diff(temp[1], t)-subs(symb_time, diff(L, theta[1]))-tau[1], trig),
{diff(theta[1](t), t, t), diff(theta[2](t), t, t)})
```

Which gives us the first equation as

$$\left(m_1 {r_1}^2 + I_1 + I_2\right) \frac{d^2}{dt^2}\theta_1(t) + m_2 g r_2 \cos(\theta_1(t) + \theta_2(t)) + [{l_1}^2 \frac{d^2}{dt^2}\theta_1(t) + {r_2}^2 \left(\frac{d^2}{dt^2}\theta_1(t) + \frac{d^2}{dt^2}\theta_2(t)\right) - 1.0\, l_1 r_2 \sin(\theta_2(t)) \left(\frac{d}{dt}\theta_2(t)\right) \left(\frac{d}{dt}\theta_1(t) + \frac{d}{dt}\theta_2(t)\right) + l_1 r_2 \cos(\theta_2(t)) \left(\frac{d^2}{dt^2}\theta_1(t) + \frac{d^2}{dt^2}\theta_2(t)\right) - 1.0\, l_1 r_2 \sin(\theta_2(t)) \left(\frac{d}{dt}\theta_2(t)\right) \frac{d}{dt}\theta_1(t) + l_1 r_2 \cos(\theta_2(t)) \frac{d^2}{dt^2}\theta_1(t)] m_2 + [0.0] + m_2 g l_1 \cos(\theta_1(t)) + I_2 \frac{d^2}{dt^2}\theta_2(t) + m_1 g r_1 \cos(\theta_1(t)) - 1.0\, \tau_1 = 0$$

Similarly, second equation for system is obtained via the command

```
[>eq[2]:=collect(simplify(diff(temp[2], t)-subs(symb_time, diff(L, theta[2]))-tau[2], trig),
{diff(theta[1](t), t, t), diff(theta[2](t), t, t)})
```

and, we get

$$[{r_2}^2 \left(\frac{d^2}{dt^2}\theta_1(t) + \frac{d^2}{dt^2}\theta_2(t)\right) - 1.0\, l_1 r_2 \sin(\theta_2(t)) \left(\frac{d}{dt}\theta_2(t)\right) \frac{d}{dt}\theta_1(t) + l_1 r_2 \cos(\theta_2(t)) \frac{d^2}{dt^2}\theta_1(t)] m_2 + [0.0] + I_2 \frac{d^2}{dt^2}\theta_1(t) + I_2 \frac{d^2}{dt^2}\theta_2(t) - 0.50\, m_2 [-2.0\, l_1 r_2 \sin(\theta_2(t)) \left(\frac{d}{dt}\theta_1(t)\right) \left(\frac{d}{dt}\theta_1(t) + \frac{d}{dt}\theta_2(t)\right)] + m_2 g r_2 \cos(\theta_1(t) + \theta_2(t)) - 1.0\, \tau_2 = 0$$

8. Now that the E-L equations have been derived, further manipulations like collecting the terms, expanding any if required can be carried using the commands `collect` & `simplify`. Once the equations are in the appropriate form, these may be exported to MATLAB and solved numerically by solvers like ODE45 (which uses Runge-Kutta methods). A significant point to be noted here is that, to transfer this equation to MATLAB, you just need to copy the equation from MAPLE using `Ctrl+C` & paste it in MATLAB environment using `Ctrl+V`. MAPLE will take care of proper syntax management between the two computing environments.

3.2 Four-bar Mechanism

Figure 3 shows the four-bar mechanism. The geometry and inertial parameters are as shown in the figure. The four-bar mechanism is perhaps the simplest closed-loop mechanism with one degree of freedom. Hence it can have only one actuated torque $\tau_1(t)$. It is assumed to be acting in the joint with joint variable $\theta_1(t)$. We present the steps in MAPLE which can be used to obtain the equations of motion of the four-bar mechanism. The steps are described in a more concise manner as most of the commands and syntaxes used here has been explained in the previous example. The commands will be listed out along with their respective outputs.

1. Define the Kinetic and Potential energies by the command

```
[>KE:=.5*m[1]*r[1]^2*dtheta[1]^2+.5*I[1]*dtheta[1]^2+.5*I[2]*(dtheta[1]+dphi[2])^2
+.5*m[2]*(l[1]^2*dtheta[1]^2+r[2]^2*(dtheta[1]+dphi[2])^2
+2*l[1]*r[2]*cos(phi[2])*dtheta[1]*(dtheta[1]+dphi[2]))+.5*m[3]*r[3]^2*dphi[1]^2+.5*I[3]*dphi[1]^2
```

&

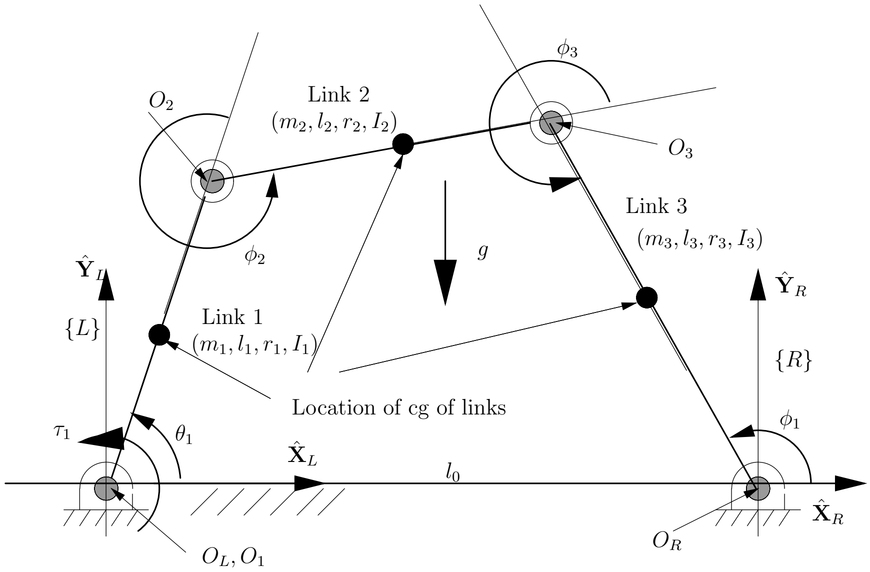

Figure 3: A four-bar mechanism

```
[>PE:=m[1]*g*r[1]*sin(theta[1])+m[2]*g*(l[1]*sin*theta[1]+r[2]*sin(theta[1]+phi[2]))
+m[3]*g*r[3]*sin(phi[1])
```

Which gives the output,

$$\begin{aligned} KE = {} & 0.5\, m_1 {r_1}^2 dtheta_1{}^2 + 0.5\, I_1\, dtheta_1{}^2 + 0.5\, I_2 \left(dtheta_1 + dphi_2\right)^2 \\ & + 0.5\, m_2 \left({l_1}^2 dtheta_1{}^2 + {r_2}^2 \left(dtheta_1 + dphi_2\right)^2 + 2\, l_1 r_2 \cos\left(\phi_2\right) dtheta_1 \left(dtheta_1 + dphi_2\right)\right) \\ & + 0.5\, m_3 {r_3}^2 dphi_1{}^2 + 0.5\, I_3\, dphi_1{}^2 \end{aligned}$$

$$PE = m_1 g r_1 \sin\left(\theta_1\right) + m_2 g \left(l_1 \sin\left(\theta_1\right) + r_2 \sin\left(\theta_1 + \phi_2\right)\right) + m_3 g r_3 \sin\left(\phi_1\right)$$

2. Therefore, the Lagrangian may be written as
```
[>L=KE-PE
```
Which gives the output as

$$\begin{aligned} L = {} & 0.5\, m_1 {r_1}^2 dtheta_1{}^2 + 0.5\, I_1\, dtheta_1{}^2 + 0.5\, I_2 \left(dtheta_1 + dphi_2\right)^2 \\ & + 0.5\, m_2 \left({l_1}^2 dtheta_1{}^2 + {r_2}^2 \left(dtheta_1 + dphi_2\right)^2 + 2\, l_1 r_2 \cos\left(\phi_2\right) dtheta_1 \left(dtheta_1 + dphi_2\right)\right) \\ & + 0.5\, m_3 {r_3}^2 dphi_1{}^2 + 0.5\, I_3\, dphi_1{}^2 - m_1 g r_1 \sin\left(\theta_1\right) - m_2 g \left(l_1 \sin\left(\theta_1\right) + r_2 \sin\left(\theta_1 + \phi_2\right)\right) - m_3 g r_3 \sin\left(\phi_1\right) \end{aligned}$$

3. Unlike the case of a 2R manipulator, where there were only active variables, there exist passive variables in this problem. The passive variables have to be eliminated using the loop closure constraint equations. In this case, the active variable is θ_1 and passive variables are ϕ_1, ϕ_2. To eliminate the passive variables, the loop closure equations to be used are,
```
[>constr[1]:=l[1]*cos(theta[1])+l[2]*cos(theta[1]+phi[2])-l[3]*cos(phi[1])-l[0]
& [>constr[2]:=l[1]*sin(theta[1])+l[2]*sin(theta[1]+phi[2])-l[3]*sin(phi[1])
```
Which gives,

$$l_1 \cos\left(\theta_1\right) + l_2 \cos\left(\theta_1 + \phi_2\right) - l_3 \cos\left(\phi_1\right) - l_0 = 0$$
$$l_1 \sin\left(\theta_1\right) + l_2 \sin\left(\theta_1 + \phi_2\right) - l_3 \sin\left(\phi_1\right) = 0$$

4. Next we calculate the terms involving derivatives with respect to rates in the E-L equations. For the first equation (with respect to θ_1)
```
[>trm[11]:=collect(simplify(diff(subs(symb_time, diff(L, dtheta[1])), t), trig), {m[1], m[2]})
```
Which in turn gives

$$\begin{aligned} & m_1 {r_1}^2 \frac{d^2}{dt^2}\theta_1(t) + \left({l_1}^2 \frac{d^2}{dt^2}\theta_1(t) - 1.0\, l_1 r_2 \sin\left(\phi_2(t)\right) \left(\frac{d}{dt}\phi_2(t)\right)^2 + {r_2}^2 \frac{d^2}{dt^2}\theta_1(t) + {r_2}^2 \frac{d^2}{dt^2}\phi_2(t) \right. \\ & \left. - 2.0\, l_1 r_2 \sin\left(\phi_2(t)\right) \left(\frac{d}{dt}\phi_2(t)\right) \frac{d}{dt}\theta_1(t) + 2.0\, l_1 r_2 \cos\left(\phi_2(t)\right) \frac{d^2}{dt^2}\theta_1(t) + l_1 r_2 \cos\left(\phi_2(t)\right) \frac{d^2}{dt^2}\phi_2(t)\right) m_2 \\ & + I_1 \frac{d^2}{dt^2}\theta_1(t) + I_2 \frac{d^2}{dt^2}\theta_1(t) + I_2 \frac{d^2}{dt^2}\phi_2(t) \end{aligned}$$

Similarly, for the corresponding term in the second equation(with respect to ϕ_1), we have

```
[>trm[12]:=collect(simplify(diff(subs(symb_time,diff(L, dphi[2])),t),trig),{m[1], m[2]})
```

This gives the output as,

$$\left(r_2{}^2 \frac{d^2}{dt^2}\theta_1(t) + r_2{}^2 \frac{d^2}{dt^2}\phi_2(t) - 1.0\, l_1 r_2 \sin(\phi_2(t)) \left(\frac{d}{dt}\phi_2(t)\right) \frac{d}{dt}\theta_1(t) + l_1 r_2 \cos(\phi_2(t)) \frac{d^2}{dt^2}\theta_1(t)\right) m_2 \\ + I_2 \frac{d^2}{dt^2}\theta_1(t) + I_2 \frac{d^2}{dt^2}\phi_2(t)$$

Finally, the same term appearing in the third equation(with respect to ϕ_2) is derived using,

```
[>trm[13]:=collect(simplify(diff(subs(symb_time,diff(L, dphi[1])), t),trig),{m[1], m[2]})
```

The term comes out as

$$1.0\, m_3 r_3{}^2 \frac{d^2}{dt^2}\phi_1(t) + 1.0\, I_3 \frac{d^2}{dt^2}\phi_1(t)$$

5. Now we move on to construct the E-L equations of motion. As already mentioned, we have three variables in this problem, as the figure indicates, namely θ_1, ϕ_1 and ϕ_2. These can be constructed as follows.

```
[>eq[1]:=collect(simplify(expand(trm[11]-subs(symb_time, diff(L, theta[1]))-tau[1], trig)),
[diff(theta[1](t), t, t), diff(phi[2](t), t, t), diff(phi[1](t), t, t), diff(phi[2](t), t),
diff(theta[1](t), t), diff(phi[1](t), t), theta[1](t), phi[1](t), phi[2](t)])
[>eq[2]:=collect(simplify(expand(trm[12]-subs(symb_time, diff(L, phi[2]))-tau[2], trig)),
[diff(theta[1](t), t, t), diff(phi[2](t), t, t), diff(phi[1](t), t, t), diff(phi[2](t), t),
diff(theta[1](t), t), diff(phi[1](t), t), theta[1](t), phi[1](t), phi[2](t)])
[>eq[3]:=collect(simplify(expand(trm[13]-subs(symb_time, diff(L, phi[1]))-tau[3], trig)),
[diff(theta[1](t), t, t), diff(phi[2](t), t, t), diff(phi[1](t), t, t), diff(phi[2](t), t),
diff(theta[1](t), t), diff(phi[1](t), t), theta[1](t), phi[1](t), phi[2](t)])
```

These three statements, upon execution give the three equations as

$$\left(m_1 r_1{}^2 + I_1 + I_2 + m_2 l_1{}^2 + m_2 r_2{}^2 + 2.0\, m_2 l_1 r_2 \cos(\phi_2(t))\right) \frac{d^2}{dt^2}\theta_1(t) + \left(I_2 + m_2 l_1 r_2 \cos(\phi_2(t)) + m_2 r_2{}^2\right) \frac{d^2}{dt^2}\phi_2(t) \\ + m_2 g r_2 \cos(\theta_1(t)) \cos(\phi_2(t)) - 1.0\, m_2 g r_2 \sin(\theta_1(t)) \sin(\phi_2(t)) - 1.0\, m_2 l_1 r_2 \sin(\phi_2(t)) \left(\frac{d}{dt}\phi_2(t)\right)^2 \\ + m_1 g r_1 \cos(\theta_1(t)) + m_2 g l_1 \cos(\theta_1(t)) - 2.0\, m_2 l_1 r_2 \sin(\phi_2(t)) \left(\frac{d}{dt}\phi_2(t)\right) \frac{d}{dt}\theta_1(t) - 1.0\, \tau_1 = 0$$

$$\left(I_2 + m_2 l_1 r_2 \cos(\phi_2(t)) + m_2 r_2{}^2\right) \frac{d^2}{dt^2}\theta_1(t) + \left(I_2 + m_2 r_2{}^2\right) \frac{d^2}{dt^2}\phi_2(t) - 1.0\, \tau_2 \\ + m_2 l_1 r_2 \sin(\phi_2(t)) \left(\frac{d}{dt}\theta_1(t)\right)^2 + m_2 g r_2 \cos(\theta_1(t)) \cos(\phi_2(t)) - 1.0\, m_2 g r_2 \sin(\theta_1(t)) \sin(\phi_2(t)) = 0$$

$$\left(m_3 r_3{}^2 + I_3\right) \frac{d^2}{dt^2}\phi_1(t) + m_3 g r_3 \cos(\phi_1(t)) - 1.0\, \tau_3 = 0$$

These are the three E-L equations for the four-bar mechanism. It may be noted $\tau_2(t)$ and $\tau_3(t)$ will be zero since there is no actuator at the passive joints. It can be non-zero if friction or some other torques are bein modeled.

6. Now, we move onto extraction of the Mass Matrix, which is obtained by collecting the coefficients of the second derivatives of the generalized coordinates(variables). This is done as follows.

```
[>M:=Matrix(3, 3, [coeff(eq[1],diff(theta[1](t), t, t)),
coeff(eq[1],diff(phi[2](t), t, t)), coeff(eq[1], diff(phi[1](t), t, t)),
coeff(eq[2], diff(theta[1](t), t, t)), coeff(eq[2], diff(phi[2](t), t, t)),
coeff(eq[2], diff(phi[1](t), t, t)), coeff(eq[3], diff(theta[1](t), t, t)),
coeff(eq[3], diff(phi[2](t), t, t)), coeff(eq[3], diff(phi[1](t), t, t))])
```

$$M = \begin{bmatrix} m_1 r_1{}^2 + I_1 + I_2 + m_2 l_1{}^2 + m_2 r_2{}^2 + 2.0\, m_2 l_1 r_2 \cos(\phi_2(t)) & I_2 + m_2 l_1 r_2 \cos(\phi_2(t)) + m_2 r_2{}^2 & 0 \\ I_2 + m_2 l_1 r_2 \cos(\phi_2(t)) + m_2 r_2{}^2 & I_2 + m_2 r_2{}^2 & 0 \\ 0 & 0 & m_3 r_3{}^2 + I_3 \end{bmatrix}$$

Further, the explicit dependence on time, of the generalized coordinates are removed using the command

```
[>M_tmp := subs({phi[1](t) = q[3], phi[2](t) = q[2], theta[1](t) = q[1]}, M)
```

Which gives the output

$$M_t mp = \begin{bmatrix} m_1 r_1{}^2 + I_1 + I_2 + m_2 l_1{}^2 + m_2 r_2{}^2 + 2.0\, m_2 l_1 r_2 \cos(q_2) & I_2 + m_2 l_1 r_2 \cos(q_2) + m_2 r_2{}^2 & 0 \\ I_2 + m_2 l_1 r_2 \cos(q_2) + m_2 r_2{}^2 & I_2 + m_2 r_2{}^2 & 0 \\ 0 & 0 & m_3 r_3{}^2 + I_3 \end{bmatrix}$$

7. Next, we extract the Coriolis/Centripetal matrix. This is constructed by collecting the coefficients of the quadratic terms. However, an alternate easier approach would be to use the formula involving the elements of the mass matrix M, which is given by

$$C_{ij} = \frac{1}{2}\sum_{k=1}^{n}\left(\frac{\partial M_{ij}}{\partial q_k} + \frac{\partial M_{ik}}{\partial q_j} - \frac{\partial M_{kj}}{\partial q_i}\right)\dot{q_k}$$

This can be implemented in MAPLE by the following code.

```
[> Ctmp := Matrix(3, 3);
for i to 3 do
 for j to 3 do
  Ctmp[i, j]:=0;
  for k to 3 do
   tmp_coeff:=subs({q[1] = theta[1](t), q[2] = phi[2](t), q[3] = phi[1](t)},
   .5*(diff(M_tmp[i, j], q[k])+diff(M_tmp[i, k], q[j])-(diff(M_tmp[k, j], q[i]))));
   Ctmp[i, j]:=Ctmp[i, j]+tmp_coeff*(diff(q[k](t), t))
  end do
 end do
end do;
C:=subs({q[1](t) = theta[1](t), q[2](t) = phi[2](t), q[3](t) = phi[1](t)}, Ctmp);
```

The above set of commands gives the C matrix as

$$C = \begin{bmatrix} -1.0\, m_2 l_1 r_2 \sin(\phi_2(t)) \frac{d}{dt}\phi_2(t) & -1.0\, m_2 l_1 r_2 \sin(\phi_2(t)) \frac{d}{dt}\theta_1(t) - 1.0\, m_2 l_1 r_2 \sin(\phi_2(t)) \frac{d}{dt}\phi_2(t) & 0.0 \\ 1.0\, m_2 l_1 r_2 \sin(\phi_2(t)) \frac{d}{dt}\theta_1(t) & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.0 \end{bmatrix}$$

8. Next, we extract the gravity vector. This can be obtained from the expression of the Potential Energy. We have

$$G_i = \frac{\partial(PE)}{\partial q_i}$$

Using the above formula, we can compute the gravity vector using

```
[> G:=subs({phi[1]=phi[1](t), phi[2]=phi[2](t), theta[1]=theta[1](t)},
Vector(3, [diff(PE,theta[1]), diff(PE,phi[2]), diff(PE,phi[1])]))
```

The output for the above command gives the gravity vector as,

$$G = \begin{bmatrix} m_1 g r_1 \cos(\theta_1(t)) + m_2 g \left(l_1 \cos(\theta_1(t)) + r_2 \cos(\theta_1(t) + \phi_2(t))\right) \\ m_2 g r_2 \cos(\theta_1(t) + \phi_2(t)) \\ m_3 g r_3 \cos(\phi_1(t)) \end{bmatrix}$$

9. Now, we bring the constraints into picture. We develop the constraint matrix Ψ, which is the derivative of the constraint equations with respect to the generalized coordinates θ_1, ϕ_1 & ϕ_2. This is done as follows.

```
[> psi := subs({phi[1] = phi[1](t), phi[2] = phi[2](t), theta[1] = theta[1](t)},
Matrix(2, 3, [[diff(constr[1],theta[1]),diff(constr[1],phi[1]), diff(constr[1],phi[2])],
[diff(constr[2],theta[1]), diff(constr[2],phi[1]), diff(constr[2],phi[2])]]))
```

The constraint matrix is obtained as follows.

$$\Psi = \begin{bmatrix} -l_1 \sin(\theta_1(t)) - l_2 \sin(\theta_1(t) + \phi_2(t)) & l_3 \sin(\phi_1(t)) & -l_2 \sin(\theta_1(t) + \phi_2(t)) \\ l_1 \cos(\theta_1(t)) + l_2 \cos(\theta_1(t) + \phi_2(t)) & -l_3 \cos(\phi_1(t)) & l_2 \cos(\theta_1(t) + \phi_2(t)) \end{bmatrix}$$

Using the constraint matrix Ψ, we can eliminate the two Lagrange multipliers which needs to be introduced for each of the constraint equations before appending them to the augmented Lagrangian $\mathcal{L}$. Once the Lagrangian multipliers are eliminated, we will be left with a set of three second-order, ordinary differential equations of motion for the planar four-bar mechanism. The steps which follow are part of simplification procedures before the system is numerically solved for some known initial conditions.

10. We know that the equations of dynamics derived using E-L equations (or by any method for that matter) can be cast in the generic form as

$$[M(q)]\ddot{q} + [C(q,\dot{q})]\dot{q} + G(q) = \tau + [\Psi(q)]^T\lambda$$

Where τ is the vector of external torques/forces, Ψ is the Jacobian of the constraints and λ's are the corresponding Lagrangian multipliers. The constraint equations will be of the form

$$[\Psi(q)]\dot{q} = 0$$

By simple algebraic manipulation, the Lagrangian multipliers can be eliminated, to arrive at the following generic form for equations of motion.

$$[M]\ddot{q} = (\tau - [C]\dot{q} - G) - [\Psi]^T([\Psi][M]^{-1}[\Psi]^T)^{-1}\left\{[\Psi][M]^{-1}(\tau - [C]\dot{q} - G) + [\dot{\Psi}]\dot{q}\right\}$$

This on further simplification yields the following expression.

$$\ddot{q} = \left([M]^{-1} - [M]^{-1}[\Psi]^T([\Psi][M]^{-1}[\Psi]^T)^{-1}[\Psi][M]^{-1}\right)(\tau - G) - [M]^{-1}[\Psi]^T([\Psi][M]^{-1}[\Psi]^T)^{-1}\left(\dot{\Psi} - \Psi[C]\right)$$

We will derive the symbolic form of various terms coming up in the above expression, so that they may be used directly in any numerical computation routine.

11. Note that from this step on, only the MAPLE commands will be provided and outputs are skipped.

To compute $[M]^{-1}$

```
[> tmp[5]:=MatrixInverse(M)
```

Next, to compute $[\Psi][M]^{-1}[\Psi]^T$ we give

```
[> tmp[1]:=MatrixMatrixMultiply(psi, MatrixMatrixMultiply(MatrixInverse(M, method = subs),
MTM[transpose](psi)))
```

To evaluate the term $[\Psi]^T$ we use MAPLE commands

```
[> tmp[3] := MTM[transpose](psi)
```

To evaluate $[\Psi][M]^{-1}\tau - [\Psi][M]^{-1}G$, we use

```
[> tmp[2]:=MatrixVectorMultiply(psi, MatrixVectorMultiply(MatrixInverse(M), T))
-MatrixVectorMultiply(psi, MatrixVectorMultiply(MatrixInverse(M), G))
```

Finally, to evaluate the term $[\dot{\Psi}] - [\Psi][M]^{-1}[C]$,

```
[> tmp[4]:=map(diff, psi, t)-MatrixMatrixMultiply(psi, MatrixMatrixMultiply(MatrixInverse(M), C))
```

12. Once the above terms are derived, the symbolic expressions may be transferred to any numercial computation environment, and the above formed system of ordinary differential equations can be solved with proper initial conditions.

4 Concluding Remarks

In the above, two examples are presented to introduce a user to MAPLE. A word of caution would be very much apt at this point. MAPLE is only a software and can never replace the logical/ analytical capabilities of a human brain. For example, MAPLE cannot easily simplify expressions by factoring out common terms, nor can it locate and remove cancellation terms. Similarly, it cannot take a symbolic inverse of too complicated a matrix, even if it exists and can be done manually. Therefore, the results should be validated by verifying the outputs for a simplified and idealized system for which the analytical results we may already know.

For example, in case of the 2R manipulator, if the second link length is made to approach zero and if the operating range of the joint angles from a vertical reference are taken to be small, then the symbolic expression for the dynamics should match with that of a simple pendulum. This can be easily verfied.