A new Maxima package for dimensional analysis

Barton Willis
University of Nebraska at Kearney
Kearney Nebraska

April 1, 2002

Introduction

This document demonstrates some of the abilities of a new Maxima package for dimensional analysis. Maxima comes with an older package dimensional analysis that is similar to the one that was in the commercial Macsyma system. The software described in this document differs greatly from the older one.

The new dimensional analysis package was written by Barton Willis of the University of Nebraska at Kearney. It is released under the terms of the General Public License GPL. You may contact the author at `willisb@unk.edu`.

Installation

To use this package, you must first download the file `dimension.mac`; it may be found at `www.unk.edu/acad/math/people/willisb`. After downloading, copy it into a directory that Maxima can find.

Usage

To use the package, you must first load it. From a Maxima prompt, this is done using the command

```
(c1)   load("dimension.mac");
(d1)
                                dimen1.mac
```

To begin, we need to assign dimensions to the variables we want to use. Use the `qput` function to do this; for example, to declare x a length, c a speed, and t a time, use the commands

```
(c2)   qput(x, "length", dimension)$
(c3)   qput(c, "length" / "time", dimension)$
(c4)   qput(t, "time", dimension)$
```

We've defined the dimensions length and time to be strings; doing so reduces the chance that they will conflict with other user variables. To declare a dimensionless variable σ, use 1 for the dimension. Thus

```
(c5)   qput(sigma,1,dimension)$
```

To find the dimension of an expression, use the `dimension` function. For example

```
(c6)   dimension(4 * sqrt(3) /t);
```

(d6)

$$\frac{1}{time}$$

```
(c7)   dimension(x + c * t);
```

(d7)

$$length$$

```
(c8)   dimension(sin(c * t / x));
```

(d8)

$$1$$

```
(c9)   dimension(abs(x - c * t));
```

(d9)

$$length$$

```
(c10)   dimension(sigma * x / c);
```

(d10)

$$time$$

(c11) `dimension(x * sqrt(1 - c * t / x));`
(d11)

$$length$$

`dimension` applies `logcontract` to its argument; thus expressions involving a difference of logarithms with dimensionally equal arguments are dimensionless; thus

(c12) `dimension(log(x) - log(c*t));`
(d12)

$$1$$

`dimension` is automatically maps over lists. Thus

(c13) `dimension([42, min(x,c*t), max(x,c*t), x^^4, x . c]);`

(d13)

$$\left[1, length, length, length^4, \frac{length^2}{time}\right]$$

When an expression is dimensionally inconsistent, `dimension` should signal an error

(c14) `dimension(x + c);`
Expression is dimensionally inconsistent.
(c15) `dimension(sin(x));`
Expression is dimensionally inconsistent.

An *equation* is dimensionally correct when either the dimensions of both sides match or when one side of the equation vanishes. For example

(c16) `dimension(x = c * t);`
(d16)

$$length$$

(c17) `dimension(x * t = 0);`
(d17)

$$length\,time$$

When the two sides of an equation have different dimensions and neither side vanishes, `dimension` signals an error

```
(c18)   dimension(x = c);
```
Expression is dimensionally inconsistent.

The function `dimension` works with derivatives and integrals

```
(c19)   dimension('diff(x,t));
```
(d19)
$$\frac{length}{time}$$

```
(c20)   dimension('diff(x,t,2));
```
(d20)
$$\frac{length}{time^2}$$

```
(c21)   dimension('diff(x,c,2,t,1));
```
(d21)
$$\frac{time}{length}$$

```
(c22)   dimension('integrate (x,t));
```
(d22)
$$length\,time$$

Thus far, any string may be used as a dimension; the other three functions in this package, `dimension_as_list` , `dimensionless`, and `natural_unit` all require that each dimension is a member of the list `fundamental_dimensions`. The default value is of this list is

```
(c23)   fundamental_dimensions;
```
(d23)
$$[mass, length, time]$$

A user may insert or delete elements from this list. The function `dimension_as_list` returns the dimension of an expression as a list of the exponents of the fundamental dimensions. Thus

```
(c24)  dimension_as_list(x);
(d24)
```

$$[0,1,0]$$

```
(c25)  dimension_as_list(t);
(d25)
```

$$[0,0,1]$$

```
(c26)  dimension_as_list(c);
(d26)
```

$$[0,1,-1]$$

```
(c27)  dimension_as_list(x/t);
(d27)
```

$$[0,1,-1]$$

```
(c28)  dimension_as_list("temp");
(d28)
```

$$[0,0,0]$$

In the last example, ”temp” isn’t an element of `fundamental_dimensions`; thus, `dimension_as_list` reports that ”temp” is dimensionless. To correct this, append ”temp” to the list `fundamental_dimensions`

```
(c29)  fundamental_dimensions : endcons("temp", fundamental_dimensions);
(d29)
```

$$[mass, length, time, temp]$$

Now we have

```
(c30)  dimension_as_list(x);
(d30)
```

$$[0,1,0,0]$$

```
(c31)  dimension_as_list(t);
```
(d31)
$$[0,0,1,0]$$

```
(c32)  dimension_as_list(c);
```
(d32)
$$[0,1,-1,0]$$

```
(c33)  dimension_as_list(x/t);
```
(d33)
$$[0,1,-1,0]$$

```
(c34)  dimension_as_list("temp");
```
(d34)
$$[0,0,0,1]$$

To remove ”temp” from `fundamental_dimensions`, use the `delete` command

```
(c35)  fundamental_dimensions : delete("temp", fundamental_dimensions)$
```

The function `dimensionless` finds a *basis* for the dimensionless quantities that can be formed from a list of dimensioned quantities. For example

```
(c36)  dimensionless([c,x,t]);
```
Dependent equations eliminated: (1)
(d36)
$$\left[\frac{ct}{x},1\right]$$

```
(c37)  dimensionless([x,t]);
```
Dependent equations eliminated: (1)
(d37)
$$[1]$$

In the first example, every dimensionless quantity that can be formed as a product of powers of c, x, and t is a power of ct/x; in the second example, the only dimensionless quantity that can be formed from x and t are the constants.

The function `natural_unit(e,[v1,v2,...,vn])` finds powers $p_1, p_2, \ldots p_n$ such that

$$\text{dimension}(e) = \text{dimension}(v_1^{p_1} v_2^{p_2} \ldots v_n^{p_n}).$$

Simple examples are

```
(c38)  natural_unit(x,[c,t]);
```
Dependent equations eliminated: (1)

(d38)
$$[ct]$$

```
(c39)  natural_unit(x,[x,c,t]);
```
Dependent equations eliminated: (1)

(d39)
$$[x]$$

Here is a more complex example; we'll study the Bohr model of the hydrogen atom using dimensional analysis. To make things more interesting, we'll include the magnetic moments of the proton and electron as well as the universal gravitational constant in with our list of physical quantities. Let $\hbar$ be Planck's constant, e the electron charge, μ_e the magnetic moment of the electron, μ_p the magnetic moment of the proton, m_e the mass of the electron, m_p the mass of the proton, G the universal gravitational constant, and c the speed of light in a vacuum. For this problem, we might like to display the square root as an exponent instead of as a radical; to do this, set `sqrtdispflag` to false

```
(c40)  SQRTDISPFLAG : false$
```

Assuming a system of units where Coulomb's law is

$$\text{force} = \frac{\text{product of charges}}{\text{distance}^2},$$

we have

```
(c41)  qput(%hbar, "mass" * "length"^2 / "time",dimension)$
```

```
(c42)  qput(%%e, "mass"^(1/2) * "length"^(3/2) / "time",dimension)$

(c43)  qput(%mue, "mass"^(1/2) * "length"^(5/2) / "time",dimension)$

(c44)  qput(%mup, "mass"^(1/2) * "length"^(5/2) / "time",dimension)$

(c45)  qput(%me, "mass",dimension)$
(c46)  qput(%mp, "mass",dimension)$
(c47)  qput(%g, "length"^3 / ("time"^2 * "mass"), dimension)$

(c48)  qput(%c, "length" / "time", dimension)$
```

The numerical values of these quantities may defined using `numerval`. We have

```
(c49)  numerval(%%e, 1.5189073558044265d-14*sqrt(kg)*meter^(3/2)/sec)$

(c50)  numerval(%hbar, 1.0545726691251061d-34*kg*meter^2/sec)$

(c51)  numerval(%c, 2.99792458d8*meter/sec)$
(c52)  numerval(%me, 9.1093897d-31*kg)$
(c53)  numerval(%mp, 1.6726231d-27*kg)$
```

To begin, let's use only the variables $e, c, \hbar, m_e$, and m_p to find the dimensionless quantities. We have

```
(c54)  dimensionless([%hbar, %me, %mp, %%e, %c]);
```

(d54)

$$\left[\frac{m_e}{m_p}, \frac{c\hbar}{e^2}, 1\right]$$

The second element of this list is the reciprocal of the fine structure constant. To find numerical values, use `float`

```
(c55)  float(%);
```

(d55)

$$\left[5.4461699709874866 \times 10^{-4}, 137.035990744505, 1.0\right]$$

The natural units of energy are given by

(c56)
```
natural_unit("mass" * "length"^2 / "time"^2, [%hbar, %me, %mp, %%e
```

(d56)

$$\left[c^2 m_e, \frac{c^3 \bar{h} m_p}{e^2}\right]$$

Let's see what happens when we include will include μ_e, μ_p, and G. We have

(c57)
```
dimensionless([%hbar, %%e, %mue, %mup, %me, %mp, %g, %c]);
```

(d57)

$$\left[\frac{\mu_p}{\mu_e}, \frac{c^2 m_e \mu_e}{e^3}, \frac{c^2 m_p \mu_e}{e^3}, \frac{e^4 G}{c^4 \mu_e^2}, \frac{c\bar{h}}{e^2}, 1\right]$$

To find the natural units of mass, length, time, speed, force, and energy, use the commands

(c58)
```
natural_unit("mass", [%hbar, %%e, %me, %mp, %mue, %mup, %g, %c]);
```

(d58)

$$\left[m_p, \frac{c^2 m_e^2 \mu_e}{e^3}, \frac{c^2 m_e^2 \mu_p}{e^3}, \frac{G m_e^3}{e^2}, \frac{c\bar{h} m_e}{e^2}\right]$$

(c59)
```
natural_unit("length", [%hbar, %%e, %me, %mp, %mue, %mup, %g, %c]);
```

(d59)

$$\left[\frac{e^2 m_p}{c^2 m_e^2}, \frac{\mu_e}{e}, \frac{\mu_p}{e}, \frac{G m_e}{c^2}, \frac{\bar{h}}{c m_e}\right]$$

(c60)
```
natural_unit("time", [%hbar, %%e, %me, %mp, %mue, %mup, %g, %c]);
```

(d60)

$$\left[\frac{e^2 m_p}{c^3 m_e^2}, \frac{\mu_e}{e c}, \frac{\mu_p}{e c}, \frac{G m_e}{c^3}, \frac{\bar{h}}{c^2 m_e}\right]$$

(c61)
```
natural_unit("mass" * "length" / "time"^2, [%hbar, %%e, %me, %mp, %m
```

(d61)
$$\left[\frac{c^4 m_e m_p}{e^2}, \frac{c^6 m_e^3 \mu_e}{e^5}, \frac{c^6 m_e^3 \mu_p}{e^5}, \frac{c^4 G m_e^4}{e^4}, \frac{c^5 \hbar m_e^2}{e^4}\right]$$

(c62)
```
natural_unit("mass" * "length"^2 / "time"^2, [%hbar, %%e, %me, %mp,
```

(d62)
$$\left[c^2 m_p, \frac{c^4 m_e^2 \mu_e}{e^3}, \frac{c^4 m_e^2 \mu_p}{e^3}, \frac{c^2 G m_e^3}{e^2}, \frac{c^3 \hbar m_e}{e^2}\right]$$

The first element of this list is the rest mass energy of the proton.

The dimension package can handle vector operators such as dot and cross products, and the vector operators div, grad, and curl. To use the vector operators, we'll first declare them

(c63)
```
prefix(div)$
```
(c64)
```
prefix(curl)$
```
(c65)
```
infix("~")$
```

Let's work with the electric and magnetic fields; again assuming a system of units where Coulomb's law is

$$\text{force} = \frac{\text{product of charges}}{\text{distance}^2}$$

the dimensions of the electric and magnetic field are

(c66)
```
qput(e, sqrt("mass") / (sqrt("length") * "time"), dimension)$
```

(c67)
```
qput(b, sqrt("mass") / (sqrt("length") * "time"),dimension)$
```

and the units of charge density ρ and current density j are

(c68)
```
qput(rho, sqrt("mass")/("time" * "length"^(3/2)), dimension)$
```

(c69)
```
qput(j, sqrt("mass") / ("time"^2 * sqrt("length")), dimension)$
```

Finally, declare the speed of light c as

(c70)
```
qput(c, "length" / "time", dimension);
```
(d70)
$$\frac{length}{time}$$

Let's find the dimensions of $\|\mathbf{E}\|^2, \mathbf{E}\cdot\mathbf{B}, \|\mathbf{B}\|^2$, and $\mathbf{E}\times\mathbf{B}/c$. We have

```
(c71)  dimension(e.e);
```
(d71)
$$\frac{mass}{length\,time^2}$$

```
(c72)  dimension(e.b);
```
(d72)
$$\frac{mass}{length\,time^2}$$

```
(c73)  dimension(b.b);
```
(d73)
$$\frac{mass}{length\,time^2}$$

```
(c74)  dimension((e ~ b) / c);
```
(d74)
$$\frac{mass}{length^2\,time}$$

The physical significance of these quantities becomes more apparent if they are integrated over $\mathbf{R}^3$. Defining

```
(c75)  qput(v, "length"^3, dimension);
```
(d75)
$$length^3$$

We now have

```
(c76)  dimension('integrate(e.e, v));
```
(d76)
$$\frac{length^2\,mass}{time^2}$$

```
(c77)  dimension('integrate(e.b, v));
```
(d77)
$$\frac{length^2\,mass}{time^2}$$

```
(c78)  dimension('integrate(b.b, v));
```
(d78)

$$\frac{length^2\, mass}{time^2}$$

```
(c79)  dimension('integrate((e ~ b) / c,v));
```
(d79)

$$\frac{length\, mass}{time}$$

It's clear that $\|\mathbf{E}\|^2, \mathbf{E}\cdot\mathbf{B}$ and $\|\mathbf{B}\|^2$ are energy densities while $\mathbf{E}\times\mathbf{B}/c$ is a momentum density.

Let's also check that the Maxwell equations are dimensionally consistent.

```
(c80)  dimension(DIV(e)= 4*%pi*rho);
```
(d80)

$$\frac{mass^{\frac{1}{2}}}{length^{\frac{3}{2}}\, time}$$

```
(c81)  dimension(CURL(b) - 'diff(e,t) / c = 4 * %pi * j / c);
```
(d81)

$$\frac{mass^{\frac{1}{2}}}{length^{\frac{3}{2}}\, time}$$

```
(c82)  dimension(CURL(e) + 'diff(b,t) / c = 0);
```
(d82)

$$\frac{mass^{\frac{1}{2}}}{length^{\frac{3}{2}}\, time}$$

```
(c83)  dimension(DIV(b) = 0);
```
(d83)

$$\frac{mass^{\frac{1}{2}}}{length^{\frac{3}{2}}\, time}$$

Conclusion and Future directions

Algorithmically, the dimensional analysis package is straightforward; nevertheless, there are many details, such as correctly setting option variables for linsolve, that need to be tended to. Let me know when you find a bug; I'll try to fix it. There may be some operators that aren't handled; again, let me know what is missing and I'll try to fix it.

Eventually, I hope that this package will work smoothly with the new physical constants package.

I could add predefined dimensions for derived units such as momentum, charge, density, etc.; however, given the plethora of schemes for electromagnetic units, I'm hesitant to do this.

This documentation was processed by batTEX, a Maxima preprocessor for TEX. The batTEXsoftware is also available from the author's web page.