The theory, algorithms and Fortran implementation of the kinematic variable m_{T2}

Tom James
Imperial College London Department of Physics

Supervisor Dr. Jürgen Reuter
Deutsches Elektronen-Synchrotron DESY Theoretical Physics Group

September 2013

Contents

1 Introduction

m_{T2} is a kinematic mass variable (actually a function as it contains one unknown), used to give bounds on the masses of a pair (or more) of unseen particles in a particle physics event under the assumption that they decay into visible ones [2]. Often, m_{T2} is called the 'stranverse' mass, due to its usefulness in estimating the masses of invisible super symmetric (SUSY) particles produced in collisions or decays at the Large Hadron Collider (LHC) or other accelerators.

Due to this usefulness, it has been decided that m_{T2} functionality should be included into the online distributed event generator and cross-section calculator WHIZARD [4, 5], available at [3]. The hope is that including an m_{T2} calculator into WHIZARD will help in the search for invisible dark matter candidates at the LHC both at 14 TeV and the projected 33 TeV upgrade, as well as at possible future machines such as the 100 TeV Very Large Hadron Collider (VLHC).

The notation used in this report is chosen to be consistent with the Fortran implementation of m_{T2}, and the comments included in the distribution.

2 Mathematical description and example

2.1 Transverse mass

The starting point for deriving m_{T2} is with the definition of the original transverse mass m_T, as defined for events at hadron colliders with one invisible particle and a measurable missing transverse momentum. Take the decay of an invisible particle, Y into an invisible particle n_1 and one or more visible particles, which we will treat as a single particle a:

$$Y \to n_1 + a \tag{1}$$

For this situation, we can define the transverse mass m_T as

$$m_T^2 = m_a^2 + m_{n_1}^2 + 2(E_T^a E_T^{n_1} - \mathbf{p}_T^a \cdot \mathbf{p}_T^{n_1}) \tag{2}$$

where the beam is defined to be along the z direction so the transverse momenta $\mathbf{p}_T = (p_x, p_y)$. The transverse energies of each particle, E_T are given by

$$E_T^a = \sqrt{m_a^2 + |\mathbf{p}_{T_a}^2|},$$

$$E_T^{n_1} = \sqrt{m_{n_1}^2 + |\mathbf{p}_{T_{n_1}}^2|}. \tag{3}$$

To simplify the mathematics, we can also define the $2+1$ dimensional momentum vector $\alpha = (E_T, p_x, p_y)$. Now we rewrite (2) in a simpler form

$$m_T^2 = (\alpha_a + \alpha_{n_1})^2 \tag{4}$$

which must therefore satisfy the inequality

$$m_T^2 \leq (p_a + p_{n_1})^2. \tag{5}$$

where we use p for the physical 4-momentum. (5) becomes an equality if a and n_1 have equal rapidity [12].

This basic idea can now be extended to the decay of two or more unknown particles described by a total missing momentum.

2.2 Stransverse mass

Now consider a situation where two Y particles are created in an event, and simultaneously decay as in (1)

$$\begin{aligned} Y &\to n_1 + a \\ Y &\to n_2 + b. \end{aligned} \tag{6}$$

Where n_1 and n_2 are assumed to be mass degenerate with unknown mass m_n. The mass of Y, m_Y is unknown. The particles a and b represent all of the visible particles produced in each decay branch, and we shall treat them as two single particles with masses of m_a and m_b that vary between events. It is important, however, that n_1 and n_2 are the only invisible particles produced in these decays. It will be shown that it is possible to calculate both m_n and m_Y using only the transverse momentum measurements made from the visible particles, and our function m_{T2}.

m_{T2} is defined as the minimum function given by all possible partitions of the measured $\mathbf{p}_T$. It is strictly a function, not a variable, as it depends trial estimate of m_n, which we shall call μ_n. We shall also use the dummy variables $\not{\mathbf{p}}_1$ and $\not{\mathbf{p}}_2$ to parameterize our lack of knowledge about the transverse momenta of the invisible particles. Using these definitions, the kinematic constraint which must be minimised over is $\not{\mathbf{p}}_1 + \not{\mathbf{p}}_2 = \not{\mathbf{p}}$. Mathematically

$$m_{T2}^2(\mathbf{p}_T^a, \mathbf{p}_T^b, \not{\mathbf{p}}_T; \mu_n) \equiv \min_{\not{\mathbf{p}}_1 + \not{\mathbf{p}}_2 = \not{\mathbf{p}}_T} [\max\{m_T^2(\mathbf{p}_T^a, \not{\mathbf{p}}_1; \mu_n), m_T^2(\mathbf{p}_T^b, \not{\mathbf{p}}_2; \mu_n)\}]. \tag{7}$$

The transverse mass function for each decay, m_T has been defined in (3). It can be useful to rewrite (7) in $2+1$ dimensional momentum vector form, to improve mathematical simplicity, and facilitate manipulation as done with the transverse mass (4). In its simplest form

$$m_{T2}^2(\mu_n) \equiv \min_{\not{\mathbf{p}}_1 + \not{\mathbf{p}}_2 = \not{\mathbf{p}}_T} [\max\{(\alpha_{n_1} + \alpha_a)^2, (\alpha_{n_2} + \alpha_b)^2)\}]. \tag{8}$$

A more complete derivation of m_{T2} can be found elsewhere [2, 1].

The information that m_{T2} provides about events is extremely useful due to its model-independency. For any event in the form of (6) when $\mu_n = m_n$, the end point m_{T2}^{max} is

equal to m_Y. When plotting m_{T2}^{max} as a function of μ_n, there is a kink in the curve at the point where $\mu_n = m_n$. Using this method it is therefore possible to find both m_Y and m_n when unknown using m_{T2} [1]. For more details on the kink method, which can often be unreliable see [10].

3 Algorithms

All algorithms use the definition defined above of m_{T2}, however their approach to the calculation varies. Presented below are the two most useful algorithms currently available.

3.1 m_{T2} algorithm by Lester and Barr

Christopher Lester and Alan Barr's code for m_{T2} is available at the Oxbridge Kinetics Library [6]. Their basic m_{T2} algorithm numerically minimises the larger of the two transverse masses, over a two dimensional space of momentum splits. To achieve this, Lester and Barr use the C++ version of the popular numerical minimisation package *Minuit2*.

3.2 m_{T2} algorithm by Cheng and Han

The basis for Hsin-Chia Cheng and Zhenyu Han's calculation method, available at UC Davis particle physics [7], is the assumption that m_{T2} is the boundary of the mass region that is consistent with the minimal kinematic constraints from both mass shell and missing transverse momentum considerations. This has been proven in [1]. For the momentum to be physical the following equalities must hold:

$$\begin{aligned} p_{n_1}^2 = p_{n_2}^2 = \mu_n^2, \\ (p_{n_1} + p_a)^2 = (p_{n_2} + p_b)^2 = \mu_Y^2, \\ p_{n_1}^x + p_{n_2}^x = \not{p}^x, \\ p_{n_1}^y + p_{n_2}^y = \not{p}^y. \end{aligned} \tag{9}$$

And from mass shell constraints it follows that

$$E_1 = \frac{p_a^x p_{n_1}^x}{E_a} + \frac{p_a^x p_{n_1}^x}{E_a} + \frac{\mu_Y^2 - \mu_n^2 - m_a^2}{2E_a}. \tag{10}$$

Now combining these conditions we produce constraints on possible p_{n_1} and p_{n_2} values:

$$-{p_{n_1}^z}^2 = -(E_{n_1}^2 - {p_{n_1}^x}^2 - {p_{n_1}^y}^2 - \mu_n^2) \leq 0,$$

$$-{p_{n_2}^z}^2 = -(E_{n_2}^2 - {p_{n_2}^x}^2 - {p_{n_2}^y}^2 - \mu_n^2) \leq 0. \tag{11}$$

These constraints produce allowed regions of $(p_{n_1}^x, p_{n_1}^y)$ and $(p_{n_2}^x, p_{n_2}^y)$ which are enclosed by an ellipse. However, since these are both dependent on the measured missing transverse momentum, we can eliminate $p_{n_2}^x$ and $p_{n_2}^y$ to place both ellipses on the $(p_{n_1}^x, p_{n_1}^y)$ plane. Solutions therefore only occur when these two ellipses overlap and the sizes of the ellipses increase monotonically with μ_Y. So, to solve for m_{T2} we simply have to increase μ_Y until there is a tangent point between the two ellipses. To test for a tangent point we observe that the two quadratic equations that describe the ellipses can be rewritten as a single quartic equation, the number of real solutions of which can be numerically calculated using the Sturm sequence method [9]. We therefore increase μ_Y until the number of real solutions to the quartic becomes non-zero. Then numerical bisection is repeated until the point of intersection is found.

In the special case when $\mu_Y = \mu_n + m_a$ (assuming m_a is the largest of the two visible masses), the first ellipse reduces to a point and it may be located inside the second elliptical region. If this is the case then the calculation is very easy and $m_{T2} = m_a + \mu_n$. This is called the 'unbalanced configuration', as opposed to the case described above, the 'balanced configuration'.

3.3 Comparison

Although there is strong numerical agreement between Cheng and Han's algorithm and Lester and Barr's, there is considerable evidence to suggest that Cheng and Han's m_T2 algorithm is the most suitable for implementing into the WHIZARD program. On a test run of $250,000$ events under a range of trial masses, the two algorithms produced a difference of 0.1 GeV with a probability of $O(10^{-3}{\sim}10^{-4})$ and a difference of 1 GeV with a probability of $O(10^{-4}{\sim}10^{-5})$. Cheng and Han claim that for small differences between the two algorithms, theirs gives the more accurate result as verified in Wolfram Mathematica [1].

The biggest advantage of Cheng and Han's algorithm is its speed. It runs $5-9$ times faster than Lester and Barr's [1], giving it a massive advantage when a large number of m_{T2} runs need to be performed, for example in the calculation of m_{TGen}. If the eventual goal is to implement m_{TGen} support into WHIZARD, Cheng and Han's algorithm is therefore the most suitable candidate. Additionally Lester and Barr's method relies on *Minuit2* support whereas Cheng and Han's is self contained.

4 Fortran Implementation

4.1 Method

For the reasons described above, it was decided that the Cheng and Hang algorithm was the best candidate for implementing into WHIZARD and the best way to do this was with a self contained piece of code natively written in Fortran. The code consists of a number of subroutines called by the main and a module file for storing global variables. The code has the ability to detect whether the visible particles a and b are massive or massless, and in the special case in which both are massless, a slightly different, faster algorithm is used. The code itself will be available online at [3].

4.2 Results

The Fortran code has been tested at tens of thousands of points in parameter space and is consistent with Cheng and Han's previous implementation to within around a thousandth of a percent in almost every single possible event. The code runs at approximately the same speed. It has been tested with both the GNU compiler *Gfortran* and the Intel compiler *Ifort*.

The only input variables that produce a difference between the Fortran code and Cheng and Han's is for the special case when $m_a = m_b$ while simultaneously $p_a^x = p_a^y = p_b^x = p_b^y$. The cause of this discrepancy at certain symmetries in phase space is unknown and requires further investigation.

5 Outlook

5.1 WHIZARD integration

Once the Fortran implantation has been shown to be consistent with the previous m_{T2} calculators, the next step is to integrate it into the WHIZARD software. Unfortunately users of m_{T2} within WHIZARD will have to define the input values themselves as opposed reading from an event generation program such as PYTHIA, as these programs currently lack the ability to easily separate the visible particles into a and b branches, as we have in the standard definition of m_{T2}.

Another issue that could arise during the integration into WHIZARD is the use of quadruple precision floating point variables, defined in Fortran as $real*16$. The algorithm demands that certain variables in the code be defined very precisely to achieve the correct solution. Specifically, these are the co-efficients of the Sturm sequence, as well as the co-efficients of the quadratic and quartic curves that describe the allowed elliptical region. It is also very important to be able to distinguish extremely small values of these variables

from zero, else divide by zero errors are generated and the code breaks. Storing them as quadruple variables was found to fix this issue, however it may cause problems when integrating into WHIZARD. WHIZARD allows the user to set a default precision to suit their needs as well as the compiler they are using. Even though both the GNU and Intel compilers support quadruple precision, theoretically there may be some compilers that do not. The m_{T2} code would therefore not work in this instance. This is expected to be a minor issue, however, and the code has been tested with all other floating point variables using the WHIZARD kinds default, while the few variables for which it is necessary are defined explicitly as quadruple.

One additional test that would be useful would be to use real LHC results. For this, access to the data would be required but might give a better insight as to how the code works in real world situations.

5.2 M_{TGen}

Along with others, Lester and Barr have explored a number of derivative functions of m_{T2} [8]. One of the most interesting of these is m_{TGen}, which is defined as the smallest value of m_{T2} which can be obtained over all possible partitions of the initial momenta into each branch of the event. This provides a lower bound estimate of m_n. A histogram of m_{TGen} over many events should theoretically reveal edge structures, the upper endpoint of which would equate to the masses of frequently produced particles. Naively m_{TGen} can be calculated by brute force, simply testing all possible partitions of momenta and returning the smallest value. This approach would end up taking 2^F m_{T2} calculation times, where F is the number of possible partitions of momenta into each branch of the event. This high number of calculations has previously restricted the use of m_{TGen}. However, with Cheng and Han's faster algorithm and work done by Lester and Barr on calculating m_{T2} very quickly under the special case of no upstream transverse mass (no initial state radiation), the use of m_{TGen} is becoming increasingly practical. We therefore plan to add m_{TGen} support to WHIZARD in the near future, taking advantage of the Fortran code for m_{T2} already produced.

References

[1] H. Cheng and Z. Han, *Minimal Kinematic Constraints and m_{T2}. JHEP* **12** (2008) arXiv:0810.5178v2 [hep-ph].

[2] A. Barr, C. Lester and P. Stevens, *m_{T2}: the truth behind the glamour. J. Phys* **G29** (2003) 2343-2363, arXiv:0304226v1 [hep-ph].

[3] Wolfgang Kilian, Jürgen Reuter, Thorsten Ohl, The WHIZARD Event Generator, September 2013, *http://whizard.hepforge.org/*

[4] W. Kilian, T. Ohl, J. Reuter, *WHIZARD: Simulating Multi-Particle Processes at LHC and ILC* , Eur.Phys.J.C71 (2011) 1742, arXiv: 0708.4233 [hep-ph].

[5] M. Moretti, T. Ohl, J. Reuter, *O'Mega: An Optimizing matrix element generator* , LC-TOOL-2001-040-rev, arXiv: hep-ph/0102195-rev.

[6] Dr Christopher Lester, Oxbridge Kinetics Library, August 2013, *http://www.hep.phy.cam.ac.uk/ lester/mt2/index.html*.

[7] Zhenyu Han, Last Updated: May 8, 2009, *http://particle.physics.ucdavis.edu/hefti/projects/doku.php?id=wimpmass*.

[8] A. Barr and C. Lester, m_{TGen}*: mass scale measurements in pair-production at colliders*. *JHEP* **12** (2007) 102, arXiv:0708.1028v4 [hep-ph].

[9] Sturm, Jacques Charles François (1829). "Mémoire sur la résolution des équations numériques". *Bulletin des Sciences de Férussac* **11** : 419–425.

[10] A. J. Barr, B. Gripaios and C. G. Lester, JHEP **0802**, 014 (2008) [arXiv:0711.4008 [hep-ph]]

[11] J. Beringer et al. (Particle Data Group), Phys. Rev. D **86**, 010001 (2012).

[12] H. Minkowski, *The fundamental equations for electromagnetic processes in moving bodies*. *Mathematisch-Physikalische Klasse*, pp. 53-111.