# Metrics for Weapon and Load-out Comparison in PlanetSide 2

**xRETRY**
The RE4PERS [RE4]
xretry.planetside@gmail.com

**Abstract**

Damage per Second and Time to Kill are common metrics to evaluate weapon and load-out balance in PlanetSide 2. This paper reflects on these metrics and highlights their limitations. Most notably, it shows their limited applicability to the majority of the player base. A new metric, the Win Margin is introduced, which represents in many ways a superior method of weapon or load-out comparison.

# Contents

# 1 Introduction

The game PlanetSide 2 offers a wide variety of weapons and load-out options, many with its own advantages and disadvantages. Naturally, discussions about the optimal load-outs and balance issues are very common. The performance of any given load-out is highly dependent on a multitude of factors, some of which are hard quantify. Consequently, in order to arrive at a conclusion, many of the factors involved are usually disregarded, resulting in an idealisation, which is not applicable to the majority of players.

This paper gives a detailed description of the Damage per Second (DPS) and Time to Kill (TTK), which represent common metrics to compare load-outs and weapons. Furthermore, it highlights the short-comings and limitations of these methods. Finally, it introduces a new metric, the Win Margin, which is able to accurately evaluate engagements with all relevant factors involved. Therefore, the Win Margin allows to draw meaningful conclusions, valid for the whole player-base.

# 2 Common Performance Metrics

## 2.1 Damage per Second (DPS)

One common way of comparing the performance between two weapons is to look at the damage output per second (DPS). It represents the total amount of damage a weapon can deal per second, assuming every bullet deals the same damage. This allows a weapon comparison in case the damage per bullet and fire rate are not the same.

$$DPS = dmg * \frac{rpm}{60}$$

The DPS is therefore an average, disregarding the quantisation of damage into single bullets. (fig. 1)

**Limitations** In reality, the damage is dealt in steps, bullet per bullet. The DPS does not describe this behavior. For certain situations, comparing the DPS values gives a completely wrong result. (fig. 2, fig. 22)

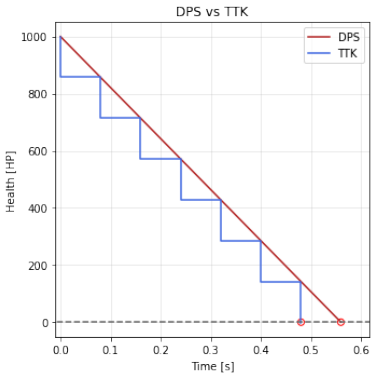



Figure 1: A direct comparison between DPS and TTK shows that, while tracking the overall trend of the real health over time, the DPS is only an approximation. Therefore, it not suited for accurate performance estimation.

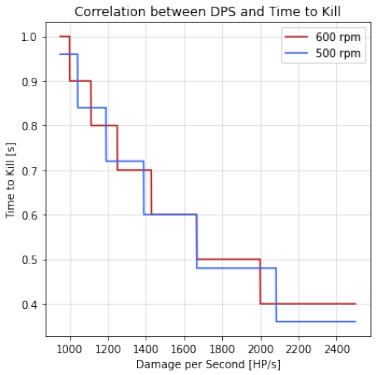



Figure 2: The correlation between Time to Kill and DPS highlights the result of damage quantisation. Every TTK has a DPS range associated with it, not a single value. Therefore, two weapons with different DPS could still have the same TTK.

## 2.2 Time to Kill (TTK)

Contrary to the DPS, the Time to Kill also takes the interaction with the target into account. Not only does this accurately describe an idealized engagement, extending the calculation to account for additional effects is intuitive and straight forward. Therefore, the TTK is a very useful metric for comparison.

### 2.2.1 Calculation

The health of the target gets divided by the bullet damage. Rounding up the result represents the hits it takes to kill the target ($htk$). Since the first bullet fires without delay, the hits to kill have to be subtracted by one and consequently multiplied by the re-fire time of the weapon.

$$TTK = (htk - 1) * t_{refire} = \left( \left\lceil \frac{health}{dmg} \right\rceil - 1 \right) * \frac{60}{rpm}$$

This formulation can be expanded to include a variety of parameters.

$$TTK = \left( \left\lceil \frac{health}{dmg_r * f_{resist}} \right\rceil - 1 \right) * \frac{60}{rpm} + t_{travel}$$

To account for distance dependent variables, the range-scaled damage is used, as well as the bullet travel time $t_{travel}$ added. Furthermore, by scaling the damage with $f_{resist}$, a target resistance has been added.

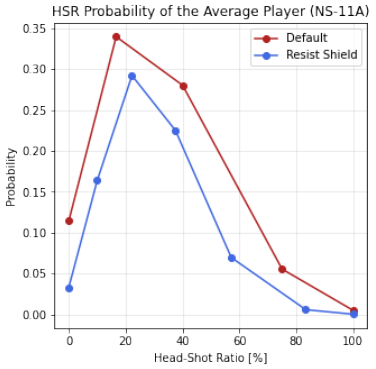



Figure 3: A player with the globally average HSR is very unlikely to hit exclusively head-shots during an engagement. This is equally valid for the accuracy.

### 2.2.2 Limitations

While effects like damage resistance and travel time can easily be accounted for, including other variables poses a significant challenge. Most notably, this simple form of the TTK is assuming that every bullet deals the same amount of damage. This is very uncommon in real engagements (fig. 3). Therefore, while being a reference, its not valid for the majority of the player-base.

# 3 Including Accuracy and Head-Shot Ratio

Most calculations of DPS and TTK do not include accuracy or head-shot ratio in any way. Therefore, they only describe a very small subset of all possible engagements. More specifically, they assume a player has perfect aim, with 100% accuracy and either 0% or 100% head-shot ratio. How a weapon performs outside of that domain remains unanswered. (fig. 4)

The addition of accuracy (ACC) and the head-shot ratio (HSR) involves overcoming a multitude of issues. The following chapter gives a detailed descriptions of the problems, as well as solutions.

## 3.1 Average Damage per Bullet

So far, when using the damage of a weapon $dmg$, it only represented a damage value which can occur in reality, either the damage of a head-shot or a body-shot. This can be generalized to the average damage per bullet $\overline{dmg}$, in which case all calculations stay the same. In this case, the value of $\overline{dmg}$ does not necessarily exist in reality, since it is the average damage, including all head-shots, body-shots and misses.

## 3.2 Damage Scale Factor (DSF)

The Damage Scale Factor describes how the damage output of a weapon changes depending on the accuracy and head-shot ratio of a player. It allows the transformation from the normal body-shot damage $dmg_n$ to the average damage $\overline{dmg}$. The DSF can either be applied to the damage by multiplication or the health by division.

$$DSF = ACC * (1 + HSR * (f_{hs} - 1))$$

Where $f_{hs}$ represents the head-shot multiplier of the weapon.

**Parallels to IvI Score** The commonly used IvI Score can be seen as a special case of the DSF, where $f_{hs} = \infty$ (fig. 5). In this case, the influence of the accuracy and head-shot ratio on the performance are weighted the same. In reality, for most infantry weapons applies $f_{hs} = 2$, therefore the IvI Score has very little explanatory power.

## 3.3 Adjusted Damage per Second

With the previously discussed DSF, the DPS can easily be adjusted to include ACC and HSR (fig. 6).

$$DPS_{adj} = \underbrace{dmg_n * DSF}_{\overline{dmg}} * \frac{rpm}{60}$$

The DSF can be thought as scaling the normal damage based on ACC and HSR.

## 3.4 Adjusted Time to Kill

Applying the DSF to the formula for the TTK results in the Adjusted TTK.

$$TTK_{adj} = \left( \left\lceil \frac{health}{dmg_n * DSF} \right\rceil - 1 \right) * t_{refire}$$

Here, the DSF can either be seen as scaling the damage or the health (fig. 7).

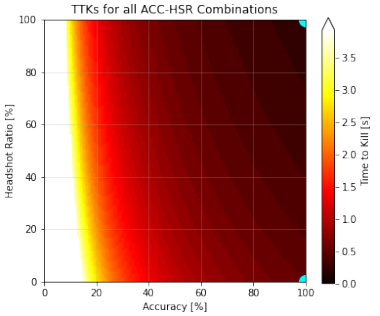



Figure 4: In this figure, the Time to Kill has been calculated for every ACC-HSR combination. Since common TTK calculations only describe the right corners of this plot (cyan points), it is unclear how a weapon would perform in all other possible situations.

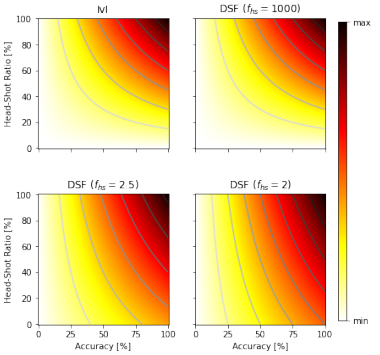



Figure 5: The IvI Score represents a special case of the DSF, where the head-shot multiplier $f_{hs} = \infty$. This is not the case for any weapon in PlanetSide. Therefore, the DSF is a better metric for quantifying the performance change due to ACC and HSR.

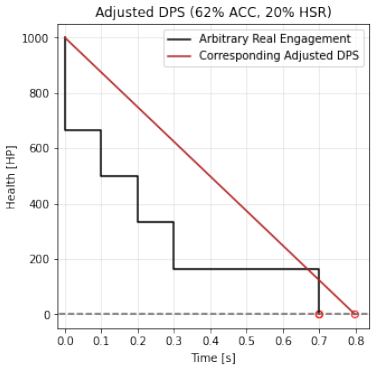



Figure 6: For a real engagement, the corresponding Adjusted DPS can be calculated. It shows that the introduction of ACC and HSR further reduces its descriptive power.

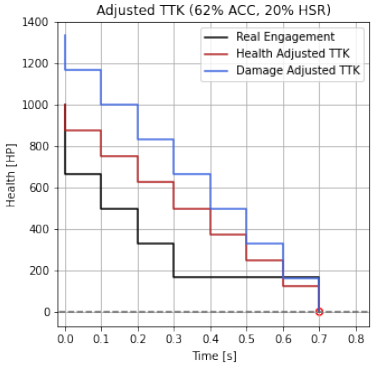



Figure 7: As highlighted in this plot, by using the Adjusted Time to Kill, it is possible to correctly calculate the TTK with ACC and HSR included. For the time-line visualization, the DSF is either applied to the health or damage. While these cases differ in the health-timeline, for the resulting TTK they are equivalent.

### 3.5 Limitations

When selecting the ACC and HSR values for the calculation, a major problem arises: Not every arbitrary value is possible in real life (fig. 8). For example, for different accuracies, the number of hits it takes to kill a target ($htk$) remains constant. What changes are the missed bullets. When seen as fraction, the denominator changes, while the nominator stays the same.

$$S_{ACC} = \left\{ \frac{htk}{htk}, \frac{htk}{htk+1}, \frac{htk}{htk+2}, ..., \frac{htk}{n_{mag}} \right\}$$

$S_{ACC}$ represents the set of all accuracy values that lead to a kill, where $n_{mag}$ is the magazine size of the weapon. When including non-kills, the nominator is replaced by hits $n_{hits}$, where $\{0 \leq n_{hits} \leq htk \mid n_{hits} \in \mathbb{N}\}$. Similarly, for all possible HSR values, $S_{HSR}$ can be represented as following set:

$$S_{HSR} = \left\{ \frac{0}{htk(0)}, \frac{1}{htk(1)}, \frac{2}{htk(2)}, ..., \frac{n_{hs}}{htk(n_{hs}) = n_{hs}} \right\}$$

In this case, $htk$ itself is dependent on the number of head-shots. More head-shots can lead to less hits being required to kill the target.

**Comparability** When comparing two players/weapons, both have their specific sets $S_{ACC}$ and $S_{HSR}$ of possible values. Using the previously presented methods, a meaningful comparison can only be done for values contained in both sets. In many cases, the number of possible comparisons is far smaller than the number of all valid ACC-HSR combinations (fig. 9).

**Summary** When using values for ACC or HSR that are not contained in their respective sets $S_{ACC}$ and $S_{HSR}$, the result can never occur in reality and it will not be meaningful in any way. Furthermore, the number of possible comparisons between two weapons/players becomes incredibly restricted. The Win Margin offers a solution to these problems.

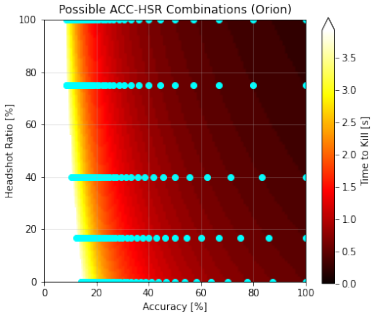



Figure 8: Not every arbitrary combinations of accuracy and head-shot ratio can occur in an real engagements. Out of the whole plane, only a small subset of points is possible (cyan).

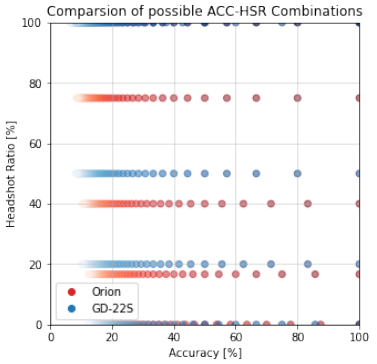



Figure 9: When comparing two weapons, the possible ACC-HSR combinations do not necessarily overlap, in which case conventional comparisons fail.

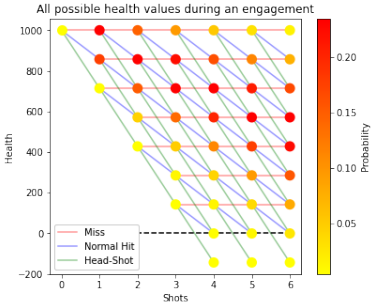



Figure 10: This figure shows the timeline of all possible health values of a player during an engagement. For a simple engagement like this, the probabilities of the health values closely resemble a tree diagram.

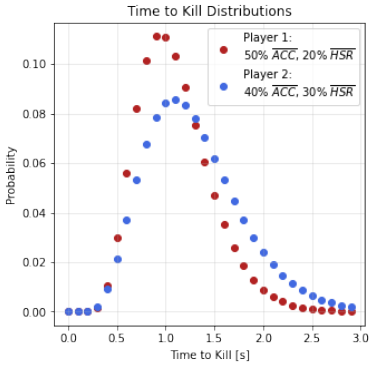



Figure 11: A Time to Kill Distribution shows every possible TTK and the corresponding probability of a player achieving it. The distribution is calculated from the players average ACC and HSR.

## 4 The Win Margin

This chapter introduces the Win Margin as a measure of performance between two weapons/players. It shows how the Win Margin, in many ways, represents a superior metric compared to TTK, while highlighting its computational challenges.

### 4.1 Introduction

Although the Adjusted TTK is able to include ACC and HSR in the calculation, it requires careful selection of those parameters. The Win Margin takes a different approach, resulting in a meaningful solution for any arbitrary value for ACC and HSR. This is achieved by using the average accuracy ($\overline{\text{ACC}}$) and head-shot ratio ($\overline{\text{HSR}}$) of a player to calculate the probability of achieving a certain ACC and HSR in a real engagement. Consequently, the performance of a player can be described by a whole TTK distribution instead of a single value (fig. 11). It assigns a probability to every possible outcome. The difference between two TTK distributions is the Win Margin (fig. 13).

### 4.2 Interpretation

The Win Margin is a statistical metric, quantifying the frequency of winning repeated engagements between two players. More specifically, assuming an infinite series of engagements, how much more does one player win over the other. It can be positive or negative, indicating which player has an advantage, with a value range between -1 and 1. A value of 1 is synonymous with a win probability of 100%.

## 4.3 Time to Kill Distribution

### 4.3.1 Concept

When using the average ACC and HSR of a player, the actual values will vary from engagement to engagement. The Adjusted TTK calculation is unable to capture this behaviour. The intuition of how to quantify this, follows the concept of a Binomial distribution, while the actual behaviour is more complex.

As described before, the accuracy is the fraction of hits out of all fired shots. Another way to look at this, is as series of independent Bernoulli trails. It can be seen as flipping a rigged coin, where the probability of success is not 50% but in this case, the accuracy. The accuracy is the average success rate of repeated shots. This concept also applies to the head-shot ratio.

A Binomial distribution describes the probability of seeing a certain outcome from a series of Bernoulli trails. Using two Binomial distributions, one for accuracy and one for head-shot ratio, it is possible to calculate the probability of achieving any ACC-HSR combination. Using this calculation alone does not give the correct result, so an additional correction needs to be applied. Considering the needed correction, as well as lacking flexibility of this calculation, it serves mainly as an intuition. A more flexible approach offers an iterative computation, which is calculating the probabilities on a step by step basis (fig. 10).

### 4.3.2 Computation

Especially when including a multitude of effects in the Win Margin calculation, this iterative approach offers a good solution. The following algorithm captures the concept, while the actual syntax and implementation varies depending on the programming language.

The algorithm computes all possible scenarios for every bullet. A bullet is equivalent to a time-step, since it can be easily converted using the re-fire and bullet-travel time. For all health values, the corresponding probabilities are calculated. Therefore, whenever a kill occurs (health is zero or smaller), the corresponding probability is known.

Using linear algebra in the computation simplifies the algorithm significantly. Therefore the initial values have to be set up as vectors. Linear algebra can also be avoided by performing the operations with multiple loops.

$$\vec{d} = \begin{bmatrix} 0 \\ dmg \\ dmg * f_{hs} \end{bmatrix}, \quad \vec{w} = \begin{bmatrix} 1 - acc \\ acc * (1 - hsr) \\ acc * hsr \end{bmatrix}, \quad \vec{h} = \begin{bmatrix} health \end{bmatrix}, \quad \vec{p} = \begin{bmatrix} 1 \end{bmatrix}$$

$\vec{d}$ contains the bullet damage for misses, normal hits and head-shots, $\vec{w}$ the corresponding probabilities. $\vec{h}$ and $\vec{p}$ contain all current health and probability values during each iteration. Therefore, they have to dynamically change in size. In general, determining the final size of $\vec{h}$ and $\vec{p}$ beforehand is not possible. If done correctly, the resulting probabilities will always sum up to one.

**Algorithm 1** Computation of a Time to Kill Distribution

**for** $i = 0 \rightarrow n_{mag}$ **do**
- $\vec{h} \leftarrow$ **flatten** $\vec{h} - \vec{d}^{\,T}$ ▷ subtracting all elements of $\vec{d}$ from every one in $\vec{h}$
- $\vec{p} \leftarrow$ **flatten** $\vec{p} * \vec{w}^{\,T}$ ▷ multiplying all elements of $\vec{w}$ with every one in $\vec{p}$
- $\vec{h}, i\vec{d}x \leftarrow$ **unique of** $\vec{h}$ ▷ removing duplicates
- $\vec{p} \leftarrow$ **accumulate** $\vec{p}$ **with** $i\vec{d}x$ ▷ grouping $\vec{p}$ based on transformation of $\vec{h}$
- $\vec{probs}\,[i] \leftarrow$ **sum** $\vec{p}$ **where** $\vec{h} \leq 0$ ▷ extracting probability of kills
- $\vec{h} \leftarrow \vec{h}$ **where** $\vec{h} > 0$ ▷ removing health values of kills
- $\vec{p} \leftarrow \vec{p}$ **where** $\vec{h} > 0$ ▷ removing probabilities of kills

**end for**

$\vec{probs}\,[n_{mag} + 1] \leftarrow$ **sum** $\vec{p}$ ▷ saving probability of non-kills

$\vec{times}\,[n_{mag} + 1] \leftarrow 1000$ ▷ saving arbitrary high time value for non-kills

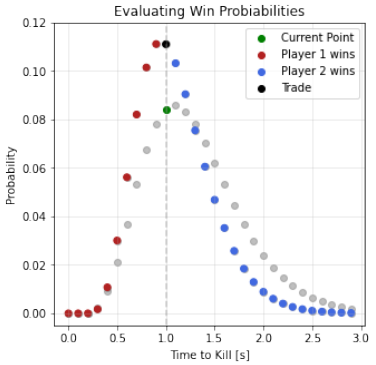



Figure 12: To calculate the win probabilities between two TTK distributions, one distribution get iterated point per point. For every point, from the other distribution the probabilities of higher and lower TTKs get summed up. They add to the final winning and losing probabilities.

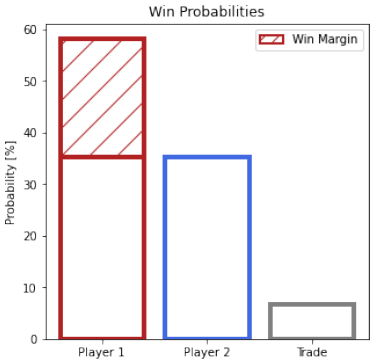
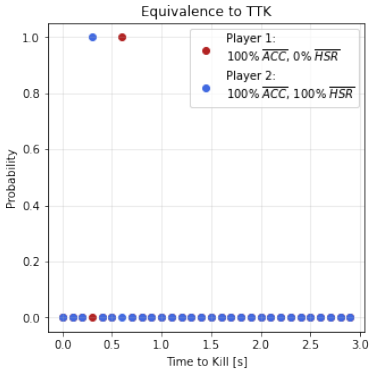



Figure 13: From two TTK distributions, the overall win probabilities for each player can be calculated. I shows how often a player would win in repeated engagements. The difference between the players win probabilities is the Win Margin.



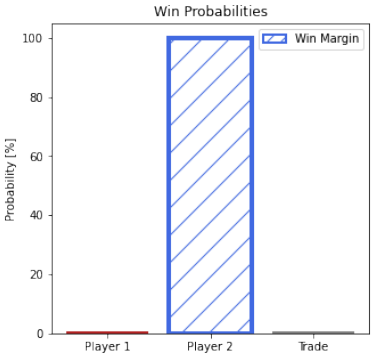




Figure 14: If a players ACC is 100% and the HSR either 0% or 100%, the TTK distribution collapses into a single value. It becomes equivalent to the common TTK calculation. The Win Margin assigns 100% win probability to the player with lower TTK, since this is the only scenario that can occur.

**Computational Challenges** In the worst case scenario, the size of $\vec{h}$ and $\vec{p}$ increases by a cubic order each iteration. This would make the computation impossible for every bullet in the magazine. In practice, this growth is severely limited or stops completely in most cases. For example, by combining equal health values, the growth of $\vec{h}$ and $\vec{p}$ is slowed down and completely stops after very few iterations. The growth is not stopped as quickly if the health is made time-dependent, which is the case when including healing or over-shield decay (fig. 16). In this case, the computational time increases significantly. This increase only becomes noticeable when performing a large amount of calculations, for example for a Win Map. A solution offers computational approximation.

## 4.4 Win Probabilities

After attaining the TTK distributions for both weapons/players, the difference between them needs to be evaluated. Computationally, for every point of the distribution, the probabilities of the other distribution get summed up, based on the relative position to the point itself. All the time-values lower than the specific point add to the losing probability, meaning the TTK of the other player would be lower. The ones higher add to the winning probability (fig. 12). Performing this calculation for each point of the distribution results in the overall probability of the player winning, losing and trading. From these three values, the most meaningful metric is the difference between winning and losing probability, which is called the Win Margin.

**Algorithm 2** Computation of a the Win Margin

**Require:** $times_{p1}$, $times_{p2}$ ▷ all possible TTKs for both players
**Require:** $probs_{p1}$, $probs_{p2}$ ▷ corresponding probabilities for both players
**Ensure:** **length** $times_{p1}$ = **length** $probs_{p1}$
**Ensure:** **length** $times_{p2}$ = **length** $probs_{p2}$
  $w_1 \leftarrow 0$ ▷ win probability of player 1
  $w_2 \leftarrow 0$ ▷ win probability of player 2
  $eq \leftarrow 0$ ▷ probability of trading kills
  **for** $i = 0 \rightarrow$ **length** $times_{p1}$ **do**
    $w_1 \leftarrow probs_{p1}[i] * \textbf{sum}\,(probs_{p2}$ **where** $times_{p2} > times_{p1}[i])$
    $w_2 \leftarrow probs_{p1}[i] * \textbf{sum}\,(probs_{p2}$ **where** $times_{p2} < times_{p1}[i])$
    $eq \leftarrow probs_{p1}[i] * \textbf{sum}\,(probs_{p2}$ **where** $times_{p2} == times_{p1}[i])$
  **end for**
  $winmargin \leftarrow w_1 - w_2$

## 4.5 Win Map

A Win Map offers an overview of the Win Margin for every ACC/HSR combination (fig. 17). It allows a qualitative comparison of the performance between two players/weapons at every skill level. Each point usually refers to both players average ACC and HSR, in which case they are assumed to be the same.

When comparing two weapons, one might be easier to control than the other. Given the same skill level or player, the average ACC and HSR of the specific weapon would be different from the overall average. This can also be included in the Win Map by introducing a weapon specific offset for the ACC and HSR used in the Win Margin computation. In this case, the overall meaning stays the same.

## 4.6 Computational Approximation

Especially when repeating the Win Margin computation many times, for example for a Win Map, it can be very computationally expensive. To improve the performance, it is possible to approximate the result.

When computing the TTK distribution, low probabilities can be disregarded, since their impact on the Win Margin is negligible (fig. 18). This can be implemented by the use of a cutoff probability in the algorithm.

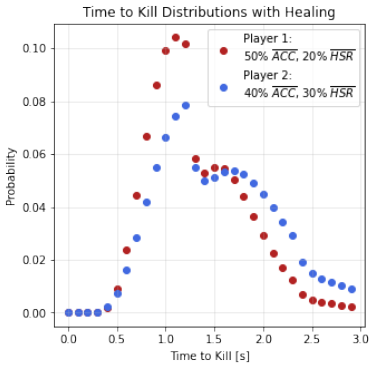



Figure 15: The addition of a healing effect heavily alters the resulting TTK distribution. The parallels to a binomial distribution mostly disappear.

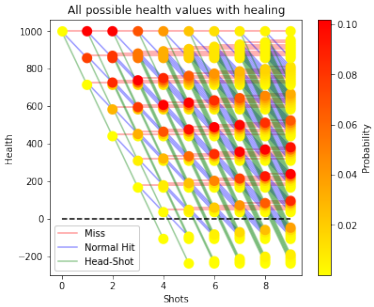



Figure 16: When healing is introduced, the amount of possible health values per shot increases significantly.

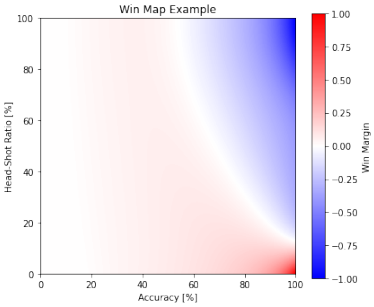



Figure 17: This figure shows an example of a Win Map. Every point in this map compares two weapons with the same average ACC and HSR. In this example, which weapon is better performing is highly depending on these values. Therefore, the Win Map gives an overview of a weapons performance based on skill level.

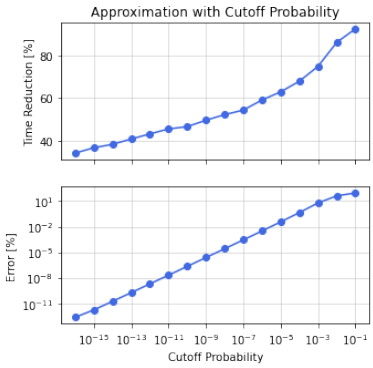



Figure 18: By introducing a cutoff probability to the computation, the time complexity can be improved. Depending on the value of this cutoff, a certain error is introduced. The result represents an approximation.

In a Win Map, due to the underlying properties, the values change smoothly throughout the map. Therefore, not every single point has to calculated. By only calculating the Win Margin for a small, evenly spaced proportion of ACC-HSR combinations, the full map can then be interpolated. This leads to a significant computational speed-up (fig. 19, fig. 20).

### 4.7 Flexibility

The main strength of the Win Margin is its ability to account for all relevant variables when comparing two weapons/players. Especially using the presented iterative computation, following effects can be included:

| Effect | Implementation |
|---|---|
| Resist Values | Scaling the damage |
| Distance | Scaling the damage and adding bullet travel time |
| ACC and HSR | Adjusting probabilities of misses, hits and head-shots |
| Latency | Adding time tolerance to Win Margin computation |
| Frame Rate | Scaling the fire rate, adding latency |
| Healing | Adding health during shots |
| Over-Shield Decay | Reducing over-shield during shots |
| Time Dependency of Effects | Changing parameters during iteration |

## 5 Conclusion

This paper has presented common metrics to compare weapon/load-out performances and highlighted their limitations. Furthermore, it introduced a new metric, the Win Margin, which solves most problems of other methods and is widely applicable. It paves the way for meaningful analysis of weapon balance and load-out options in PlanetSide 2.

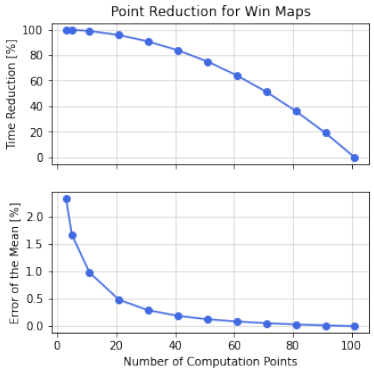



Figure 19: When creating a Win Map, it is not necessary to compute the Win Margin for every single point. Instead, only a small subset has to be calculated and the map can be interpolated.

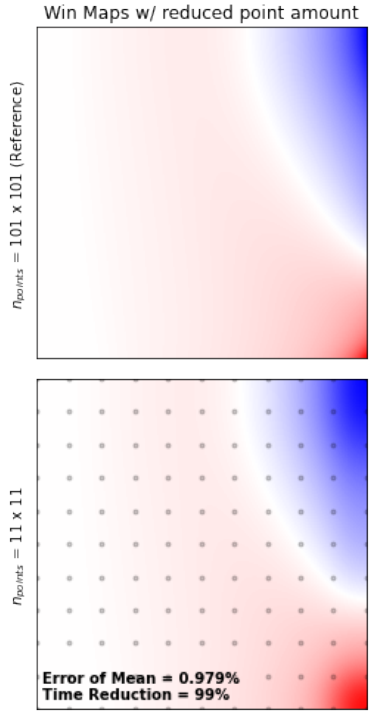



Figure 20: An interpolated Win Map with less computation points (grey points) is very similar to the original, full map. The computational speed-up is significant.

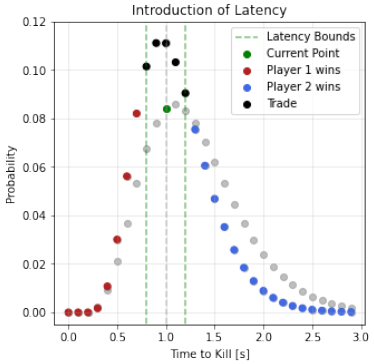



Figure 21: To include the effect of latency, a time interval can be added to the win probability computation. It increases the amount of trades.

# 6 Appendix

## 6.1 Units

| Unit | Description |
|---|---|
| $s$ | Seconds |
| $min$ | Minutes |
| $m$ | Meters |
| $hp$ | Health Points |
| $bul$ | Bullets |

## 6.2 Variables

| Variable | Unit | Description |
|---|---|---|
| $ACC$ | - | Accuracy |
| $d_{max}$ | $hp$ | Maximum bullet damage |
| $d_{min}$ | $hp$ | Minimum bullet damage |
| $dmg$ | $hp$ | Bullet damage |
| $\overline{dmg}$ | $^{hp}/_{bul}$ | Average bullet damage |
| $dmg_{hs}$ | $hp$ | Damage of a head-shot |
| $dmg_n$ | $hp$ | Damage of a normal hit (torso) |
| $dmg_r$ | $^{hp}/_{bul}$ | Range adjusted bullet damage |
| $DPS$ | $^{hp}/_{s}$ | Damage per second |
| $f_{hs}$ | - | Head-shot multiplier |
| $f_{resist}$ | - | Damage resistance factor |
| $health$ | $hp$ | Target health |
| $HSR$ | - | Head-shot ratio |
| $htk$ | $bul$ | Hits to kill the target |
| $n_{hits}$ | $bul$ | Total amount of hits |
| $n_{hs}$ | $bul$ | Amount of hits with head-shot damage |
| $n_{mag}$ | $bul$ | Magazine size |
| $n_n$ | $bul$ | Amount of hits with normal damage |
| $n_{shots}$ | $bul$ | Total amount of shots |
| $r$ | $m$ | Engagement distance |
| $r_{max}$ | $m$ | Maximum damage range |
| $r_{min}$ | $m$ | Minimum damage range |
| $rpm$ | $^{bul}/_{min}$ | Fire rate |
| $TTK$ | $s$ | Time to kill |
| $t_{refire}$ | $s$ | Weapon re-fire time |
| $t_{travel}$ | $s$ | Bullet travel time |
| $v_{bul}$ | $^{m}/_{s}$ | Bullet velocity |

## 6.3 Formulas and Derivations

### 6.3.1 Range Adjusted Damage

$$dmg_r(r) = d_{max} + \frac{d_{min} - d_{max}}{r_{min} - r_{max}} * (r - r_{max})$$

$$unit = \frac{hp}{bul} + \frac{\frac{hp}{bul} - \frac{hp}{bul}}{m - m} * (m - m) = \frac{hp}{bul}$$

### 6.3.2 Bullet Travel Time

$$t_{travel} = \frac{1}{v_{bul}} * dis$$

$$unit = \frac{1}{\frac{m}{s}} * m = \frac{s}{m} * m = s$$

### 6.3.3 Damage per Second (DPS)

$$DPS = dmg * \frac{rpm}{60}$$

$$unit = \frac{hp}{bul} * \frac{\frac{bul}{min}}{\frac{s}{min}} = \frac{hp}{bul} * \frac{bul}{s} = \frac{hp}{s}$$

### 6.3.4 Time to Kill (TTK)

$$TTK = \left( \left\lceil \frac{health}{dmg} \right\rceil - 1 \right) * \frac{60}{rpm}$$

$$\begin{aligned} unit &= \left( \left\lceil \frac{hp}{\frac{hp}{bul}} \right\rceil - bul \right) * \frac{\frac{s}{min}}{\frac{bul}{min}} = \\ &= (\lceil bul \rceil - bul) * \frac{s}{bul} = \\ &= s \end{aligned}$$

### 6.3.5 Damage Scale Factor (DSF)

The goal of this derivation is to find the average damage, or damage per shot, depending on accuracy $ACC$ and head-shot ratio $HSR$. The derivation shows that the result separates into the normal bullet damage multiplied by an expression, the DSF

$$\begin{aligned} HSR &= \frac{n_{hs}}{n_{hits}} \\ &\implies n_{hs} = HSR * n_{hits} \\ n_{hits} &= n_n + n_{hs} = n_n + HSR * n_{hits} \\ &\implies n_n = n_{hits} * (1 - HSR) \end{aligned}$$

$$\begin{aligned} \overline{dmg} &= \frac{dmg_n * n_n + dmg_{hs} * n_{hs}}{n_{shots}} = \\ &= \frac{dmg_n * n_n + dmg_n * f_{hs} * n_{hs}}{n_{shots}} = \\ &= dmg_n * \frac{n_{hits} * (1 - HSR) + f_{hs} * HSR * n_{hits}}{n_{shots}} = \\ &= dmg_n * \underbrace{\frac{n_{hits}}{n_{shots}}}_{ACC} * (1 - HSR + f_{hs} * HSR) = \\ &= dmg_n * \underbrace{ACC * \big(1 + HSR * (f_{hs} - 1)\big)}_{DSF} \end{aligned}$$

$$DSF = ACC * \big(1 + HSR * (f_{hs} - 1)\big)$$

## 6.4 Example

### 6.4.1 Overview

| | NS-11C (SPA) | Pulsar C (HVA) | | |
|---|---|---|---|---|
| Variable | Weapon 1 | Weapon 2 | Unit | Description |
| $d_{max}$ | 143 | 167 | $hp/bul$ | Maximum Damage |
| $d_{min}$ | 112 | 125 | $hp/bul$ | Minimum Damage |
| $r_{max}$ | 15 | 8 | $m$ | Max. Damage Range |
| $r_{min}$ | 60 | 90 | $m$ | Min. Damage Range |
| $rpm$ | 652 | 577 | $bul/min$ | Fire Rate |
| $v_{bul}$ | 477 | 567 | $m/s$ | Muzzle Velocity |
| $f_{hs}$ | 2 | 2 | - | Head-Shot Multiplier |
| $dis$ | 11 | 11 | $m$ | Distance |
| $ACC$ | 100 | 100 | % | Accuracy |
| $HSR$ | 0 | 0 | % | Head-Shot Ratio |
| $dmg_r$ | 143 | 165.5 | $hp/bul$ | Range Adjusted Damage |
| $t_{travel}$ | 0.0231 | 0.0194 | $s$ | Bullet Travel Time |
| ***DPS*** | 1551 | **1588** | $hp/s$ | Damage per Second |
| ***TTK*** | **0.575** | 0.643 | $s$ | Time to Kill |
| ***WM*** | **100** | - | % | Win Margin |
| $ACC$ | 60 | 40 | % | Accuracy |
| $HSR$ | 30 | 90 | % | Head-Shot Ratio |
| $DSF$ | 0.78 | 0.76 | – | Damage Scale Factor |
| ***DPS*** | **1209** | 1207 | $hp/s$ | Damage per Second |
| ***TTK*** | 0.759 | **0.747** | $s$ | Time to Kill |
| ***WM*** | **26.3** | - | % | Win Margin |

### 6.4.2 Range dependent Variables

$$dmg_{ns} = d_{max} = 143\ {}^{hp}/_{bul}$$
$$dmg_{pu} = dmg_r(11) = 167 + \frac{125-167}{90-8} * (11-8) = 165.5\ {}^{hp}/_{bul}$$
$$t_{travel,ns} = \frac{1}{477} * 11 = 0.0231\ s$$
$$t_{travel,pu} = \frac{1}{567} * 11 = 0.0194\ s$$

### 6.4.3 Damage per Second (range adjusted)

$$DPS_{ns} = 143 * \frac{652}{60} - \frac{143}{0.0231} = 1551\ {}^{hp}/_{s}$$
$$DPS_{pu} = 165.5 * \frac{577}{60} - \frac{165.5}{0.0194} = 1588\ {}^{hp}/_{s}$$

### 6.4.4 Time to Kill

$$htk_{ns} = \left\lceil \frac{1000}{143} \right\rceil = \lceil 6.993 \rceil = 7\ bul$$
$$htk_{pu} = \left\lceil \frac{1000}{165.5} \right\rceil = \lceil 6.044 \rceil = 7\ bul$$
$$TTK_{ns} = (7-1) * \frac{60}{652} + 0.0231 = 0.575\ s$$
$$TTK_{ns} = (7-1) * \frac{60}{577} + 0.0194 = 0.643\ s$$

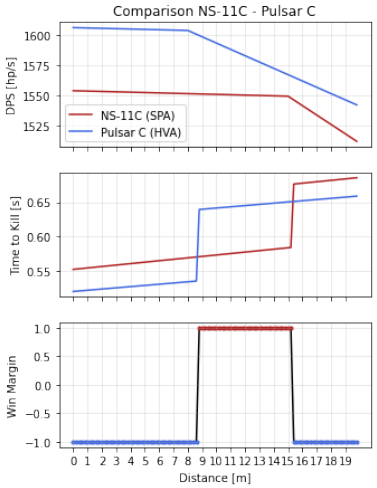



Figure 22: For the example at hand, the DPS predicts a completely wrong result at a distance between 8 to 15m. On the other hand, the TTK and Win Margin correctly predict the winner. The Win Margin jumps between 100% probability for one weapon to the other. This is due to the 100% ACC and 0% HSR in this example. There is no variation between engagements, therefore the better weapon will win 100% of the time.

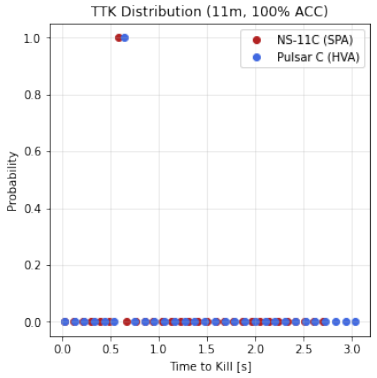



Figure 23: The TTK Distributions for the example with 100% accuracy spikes at one value for each weapon, the same value as predicted from the normal TTK calculation.

### 6.4.5 Damage Scale Factor (DSF)

$$DSF_{ns} = 0.6 * \big(1 + 0.3 * (2-1)\big) = 0.78$$
$$DSF_{pu} = 0.4 * \big(1 + 0.9 * (2-1)\big) = 0.76$$

### 6.4.6 Adjusted Damage per Second

With the DSF the DPS can be ACC & HSR adjusted but as shown before, the result is very unreliable.

$$DPS_{ns} = 143 * 0.78 * \frac{652}{60} - \frac{143 * 0.78}{0.0231} = 1209\ {}^{hp}/_{s}$$
$$DPS_{pu} = 165.5 * 0.76 * \frac{577}{60} - \frac{165.5 * 0.76}{0.0194} = 1207\ {}^{hp}/_{s}$$

### 6.4.7 Adjusted Time to Kill

In this case, the ACC and HSR values for this calculation are not possible in real life. The TTK can still be calculated but the result is completely useless.

$$htk_{ns} = \left\lceil \frac{1000}{143 * 0.78} \right\rceil = \lceil 8.965 \rceil = 9\ bul$$
$$htk_{pu} = \left\lceil \frac{1000}{165.5 * 0.76} \right\rceil = \lceil 7.950 \rceil = 8\ bul$$
$$TTK_{ns} = (9-1) * \frac{60}{652} + 0.0231 = 0.759\ s$$
$$TTK_{ns} = (8-1) * \frac{60}{577} + 0.0194 = 0.747\ s$$

### 6.4.8 Summary

The example shows that in certain cases, DPS and TTK completely fail to describe reality. The Win Margin on the other hand, even though harder to compute, can still reach a meaningful solution.

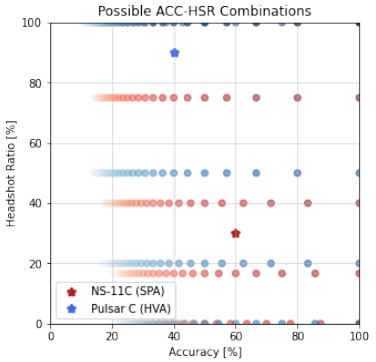



Figure 24: In the example where both weapons are compared with different ACC and HSR, only the Win Margin provides a meaningful result. The results of the other methods have no relation to the real world.

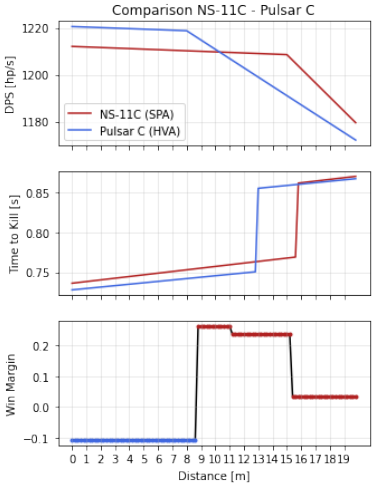



Figure 25: With included ACC and HSR values, the DPS and TTK can still be calculated, but the result becomes useless.

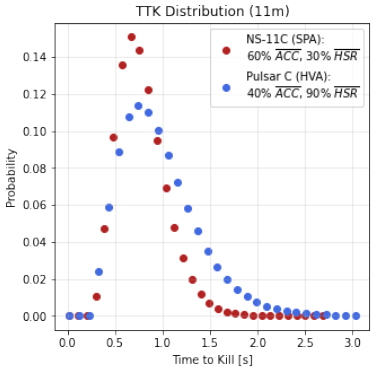



Figure 26: The TTK distributions for the example with different ACC and HSR values.

## 6.5 Code Implementations

### 6.5.1 Python

```
import numpy as np

def ttk_dist(h_init, dmg, f_hs, n_mag, rpm, acc, hsr):
    times = np.arange(n_mag+1)*60/rpm
    probs = np.zeros([n_mag+1])
    d = np.array([0, dmg, dmg*f_hs])[:, None]
    w = np.array([
        1-acc,
        acc*(1-hsr),
        acc*hsr
    ])[:, None]
    h = np.array([h_init])
    p = np.array([1])
    for s in range(n_mag):
        h = (h - d).flatten()
        p = (p * w).flatten()
        h, idx = np.unique(h, return_inverse=True)
        p = np.bincount(idx, weights=p)
        h_zero = h <= 0
        probs[s] = p[h_zero].sum()
        h = h[h_zero == False]
        p = p[h_zero == False]
    probs[-1] = p.sum()
    times[-1] = 1000
    return times, probs
```

```
def win_margin(times1, probs1, times2, probs2):
    w1 = w2 = eq = 0
    for i in range(len(times1)):
        w2 += probs1[i] * probs2[times2 < times1[i]].sum()
        w1 += probs1[i] * probs2[times2 > times1[i]].sum()
        eq += probs1[i] * probs2[times2 == times1[i]].sum()
    return w1 - w2
```

### 6.5.2 Java

```
public static double[][] ttk_dist(int n_mag, double h_init,
double dmg, double f_hs, double rpm, double acc, double hsr) {
    double[][] dist = new double[2][n_mag+1];
    double[] w = {
        1-acc,
        acc*(1-hsr),
        acc*hsr
    };
    double[] d = {0, dmg, dmg*f_hs};
    HashMap<Double, Double> h = new HashMap<Double, Double>();
    h.put(h_init, 1.);
    for (int s=0; s<n_mag; s++) {
        dist[0][s] = s*60/rpm;
        HashMap<Double, Double> h_new = new HashMap<Double, Double>();
        for (Map.Entry<Double, Double> cur : h.entrySet()) {
            for (int j=0; j<3; j++) {
                double h_cur = cur.getKey() - d[j];
                Double p_cur = cur.getValue() * w[j];
                if (h_cur <= 0) {
                    dist[1][s] += p_cur;
                    continue;
                }
                if (h_new.containsKey(h_cur)) {
                    h_new.put(h_cur, h_new.get(h_cur)+p_cur);
                } else {
                    h_new.put(h_cur, p_cur);
                }
            }
        }
        h = h_new;
    }
    for (Map.Entry<Double, Double> cur : h.entrySet()) {
        dist[1][dist[1].length-1] += cur.getValue();
    }
    dist[0][dist[0].length-1] = 1000;
    return dist;
}
```

```
public static double win_margin(double[][] dist1, double[][] dist2) {
    double[] winProbs = new double[3];
    for (int i=0; i<dist1[0].length; i++) {
        double sumEq, sumP1, sumP2;
        sumEq = sumP1 = sumP2 = 0;
        for (int j=0; j<dist1[0].length; j++) {
            if (dist2[0][j] > dist1[0][i]) sumP1 += dist2[1][j];
            else if (dist2[0][j] < dist1[0][i]) sumP2 += dist2[1][j];
            else sumEq += dist2[1][j];
        }
        winProbs[0] += dist1[1][i] * sumP1;
        winProbs[1] += dist1[1][i] * sumP2;
        winProbs[2] += dist1[1][i] * sumEq;
    }
    return winProbs[0] - winProbs[1];
}
```