lithium

`lithium` is a battery capacity optimiser for drone builds that uses machine learning techniques to find the optimum battery capacity for a drone.

flight time calculation

> *This formula is designed to observe how different batteries affect the total flight time of a drone. Because of this, it is assumed that the capacity and voltage of the battery is known. Likewise, the formula assumes that the following information is already known about the motor:*
>
> - *Maximum current draw under load*
> - *Maximum power conusmption under load*

Battery capacity is measured in amp-hours, where a 2Ah battery can power a system drawing 2A for 1 hour, or a system drawing 1A for 30 min, and so on. As you can see, the relationship between capacity and usage time is a division, where the usage time (in hours) is equal to the capacity divided by the current draw. Applying this formula to drones. we can say:

$$t = \frac{c}{\mathrm{ACD}}$$

> *Where:*
>
> - t *is time (**hours**)*
> - ACD *is Average Current Draw (**A**)*
> - c *is the capacity of the battery (**Ah**)*

However, for safety reasons, LiPo batteries should never be discharged below 80%, which reduces our usable capacity. This percentage can be tweaked depending on how much risk you're willing to take with your battery (but you really shouldn't), so we'll add it into the equation:

$$t = \frac{c \cdot D}{\mathrm{ACD}}$$

> *Where D is discharge **percentage in decimal form**.*

Next, we have to calculate average current draw. ACD consists of a varaible part I_v, and a constant part, I_c. For the constant part, I_c, we simply take a sum of the current draw of all components that will consistently draw the same current, regardless of the amount of thrust being applied.

The variable part, I_v represents the current drawn by the motors. This which will change depending on how much thrust is being applied which is why we take an *average* for these calculations. To find a value for the average current drawn by a single motor, I, we use the equation for electrcal power:

$$I \cdot V = P$$

$$I = \frac{P}{V}$$

> *Where:*
>
> - P *is power consumption of the motor under (**W**)*
> - V *is the voltage being applied to the motor (**V**)*

Considering we already know V, we have two ways to find I. Either find P and solve the equation for I, or ingore the equation and find I from some other current draw value.

Since we are looking for *average* current draw, we'll take P to be the *average* power consumption of one motor over the entire flight. This can easily be found by taking some percentage of the maximum power consumption, P_{MAX}, assuming that P will be around that percentage of P_{MAX} over the course of the flight. We'll call this ratio *"Flight Intensity"*, F - which will be a percentage in decimal form. Using this we find:

$$P = F \cdot P_{MAX}$$

$$I_1 = \frac{P}{V}$$

$$I_1 = \frac{F \cdot P_{MAX}}{V}$$

For our second method, ignoring the equation, we can find I by taking a percentage of the maximum current draw, I_{MAX} of the motors. This percentage will be our *Flight Intensity* from last time, on the same assumption that over the course of the whole flight, the average current draw, I_2 will be $F \cdot I_{MAX}$.

To find I_v, we multiply both of our expressions for I by the number of motors, N, and now we can create expressions for ACD:

$$\mathrm{ACD} = I_c + I_v$$

$$\text{ACD} = I_c + N \cdot I_1 = I_c + \frac{N \cdot F \cdot P_{MAX}}{V}$$

$$\text{ACD} = I_c + N \cdot I_2 = I_c + N \cdot F \cdot I_{MAX}$$

This means we have two forms of our flight time equation.

Power Consumption Form:

$$t_1 = \frac{c \cdot D}{I_c + \frac{N \cdot F \cdot P_{MAX}}{V}}$$

Current Draw Form:

$$t_2 = \frac{c \cdot D}{I_c + N \cdot F \cdot I_{MAX}}$$

Finally, we need to find F - the Flight Intensity. This will be a multiplier that we'll apply to the maximal power consumption and current draw of a motor, to estimate the average draw/consumption over the whole flight. To achieve this, we'll let F be decimal between 0 and 1, where $F = 0$ is the flight intensity of a drone that requires no thrust to overcome the weight of the drone and $F = 1$ is the flight intensity of a drone that requires the entire thrust capacity to match the weight of the drone. All that's left is to find our source for the values of F.

Thrust to weight ratio (TWR) is a value commonly used when desinging drones and represents the amount of thrust that the drone is capable of producing, in proportion to the weight that the thrust has to displace. The higher the thrust to weight ratio is, the easier it will be for the motors to overcome the weight of the drone, making the drone more manouverable as there is lots of headroom over the hovering point of the motors. If the thrust to weight ratio is too low, the motors will struggle to generate enough thrust to overcome the weight of the drone. This will result in the motors needing to be run at higher thrust, drawing more current and consuming more power.

This is useful for our situation, since F represents the proportion of some the greatest draw/consumption of the motor and TWR represents the proportion of thrust generated by the motor. If the TWR of the drone is high, F will be low because the motors can overcome the weight of the drone without much effort, but when TWR is lower, F will be closer to 1 because the motors will need to generate much more thrust, consming more power and drawing more current. To relate these two, we'll take F as the reciprocal of TWR. This not only accurately models the desired relationship of F to TWR, but also gives us a value between 0 and 1 since $TWR > 1$ for any drone that files.

TWR is calculated by $\frac{T}{W}$ where T is the total thrust capacity of the drone in newtons and W is the weight of the drone in newtons. Both of these are forces, making them vector

values. We tale F as TWR^{-1}, so $F = fracWT$.

Unfortunately, it won't be that simple. Motor manufacturers rarely state the actual thrust (in newtons) generated by the motor, instead opting to write "thrust tables", which tell consumers the *"pull"* generated by a motor, normally ***in grams*** (this is NOT thrust). Pull is measured by spinning the motor at full thrust against a scale, and measuring the read-out given by the scale. This read-out is the scalar component (mass) of the force (thrust) being generated by the motor. The vector component, is it's acceleration, which is the negative of the gravitational field strength *(since the drone is a free-falling object subject to gravity, and the motors are acting against gravity)*.

We know the acceleration of W, which is the gravitational field strength - because the drone is a free-falling object subject to gravity. From this, we see that the magnitude of the vector components are equal for both forces, meaning the division will cause the accelerations to cancel out, so we can find F using only the scalar components. We could have completed the calculation for F using vectors, but if we can simply the expression into one dimension, we might as well. This leaves us with:

$$F = \frac{m}{N \cdot p}$$

> Where:
> m is the total mass of the drone
> N is the number of motors on the drone
> p is the pull value of each motor

As mentioned earlier, this expression for F is only a baseline, different builds may run at different average thrusts and the expression for F needs to be able to account for this. To do this, we'll introduce a value, b to act as a bias and add it to our current expression for F. The desired effect of the bias is to offset the resultant value in a particular direction by some magnitude. Note how since F represents a proportion of the maximal values of the motors, any $F > 1$ or < 0 is invalid, so these bounds define the limits of our model.

$$F = \frac{m + 0.005b}{N \cdot p}$$

As you can see, the introduction of b makes F more adaptable, allowing the value to be offset, but also allowing remain F to remain general if $b = 0$. The 0.005 multiplier applied to b is there to soften the effect that the bias value has on the value of F.

Another thing to notice here is that fraction in F contains N in it's denominator, and in both of our forms for T, we multiply F by N. Because of this if we move the bias into the

numerator, N will cancel out in all the places F is used and we can remove it from our equations entirely. Now we have the forms for t with F:

> Power Consumption Form:
>
> $$t_1 = \frac{c \cdot D}{I_c + \frac{F \cdot P_{MAX}}{V}}$$
>
> Current Draw Form:
>
> $$t_2 = \frac{c \cdot D}{I_c + F \cdot I_{MAX}}$$
>
> Where
>
> $$F = \frac{m + 0.005b}{p}$$

Each of these forms is very simple, since we achieved our orignal goal of dividing the total usable capacity of the battery by some estimated average current draw.

Since we have two expressions for t, we will take t to be the mean average between t_1 and t_2 :

> $$F = \frac{m + 0.005b}{p}$$
>
> $$t = \frac{\frac{c \cdot D}{I_c + \frac{F \cdot P_{MAX}}{V}} + \frac{c \cdot D}{I_c + F \cdot I_{MAX}}}{2}$$
>
> $$\{c > 0\}, \{0 \leq F \leq 1\}$$

> *Where:*
>
> - m *is the total mass of the drone (**kg**).*
> - p *is the pull produced by a single motor (**kg**).*
> - b *is the **Flight Intensity bias**.*
> - t *is flight time (**hours**).*
> - c *is the capacity of the battery (**Ah**).*
> - D *is discharge **percentage in decimal form**.*
> - I_c *constant current draw (**A**).*
> - P_{MAX} *is the maximum power consumption of a single motor (**W**)*
> - V *is the voltage of the battery (**V**).*

- I_{MAX} *is the maximum current drawn by a single motor (**A**)*

The Function

This is the estimation for the CEDRIC drone's battery life model. The model was obtained by estimating the relationship between battery mass and battery capacity, which resulted in a gradient which could be used for expressing mass in terms of capaicty, leaving only one unknown variable in the expression for tm allowing the graph to be expressed in two dimensions. It is worth noting that the estimated gradient is linear but the degree of the actual realtionship between mass and capacity is unknown, so extrapolating masses from capacities outside of the original dataset used for the estimation may yield innacurate results.

The shaded area shows the parts of the graph where the drone will be too heavy to fly ($F > 0.85$). Think back to earlier when we said that the function will be bounded by F. This is beacuse F cannot be negative or greater than 1. $F < 0$ represents a physically impossible weight to thrust ratio, and when $F = 1$ the entire thrust is being used to make the drone hover. This means that if $F > 1$, the drone would be so heavy that it would require more thrust to hover than the motors are capable of producing so the drone would not be able to fly.

Therefore, the best possible battery isn't as simple as just taking the highest point on the graph for t, since at this point, $F = 1$ so the drone won't actually fly. If we say $F > 0.85$ is the range for F where the drone is too heavy to fly, the theoretically ideal battery is the point just before $F = 0.85$ on the graph. Looking deeper into this, we may want F to be even lower than this "optimal" point, since if the drone is hovering at 85% thrust, there isn't much headroom in the thrust capacity of the drone above the hovering point, meaning that it wouldn't be very manouverable, even though it *can* fly.

Ideally, we want to set a target for F representing the desired flight intensity, then solve the model to find the capacity at this flight intensity target, resulting in the optimal battery that will give the longest flight time at the given intensity. From this, it may seem that the bias is no longer necessary, since the process of setting F to a specific values looks to make the offset provided by b obselete. This is not the case, in fact, the bias is still important, since it incoporates the design purpose of the drone into the flight time model directly. For example, the model of flight time for a racing build should yield higher F values, since racing drones spend more time at higher throttle. If we ignored the bias, to we would have to set our F target to a higher value to compensate for this fact, but by utilising the bias, we incorporate the extra intensity of the flight in the model itself.

What we learn from this is that F is the main focus of our equation. We look to t to find the results of our targets, but really the F values control everything. Lets look at the graph for F to learn more:

The key values of this graph are F_{base} and $F_{gradient}$, where F_{base} is the y-intercept of the graph and $F_{gradient}$ is the gradient of the graph. These values are important because they influence the position of the F graph, so by reducing them, we move our F_{target} further along the graph, resulting in better flight times.

Once we have our chosen F_{target}, we can solve our F equation for c to find the desired capacity for this flight inensity. We don't need to put N back into our F expression despite the fact that there's nothing to cancel it out because the F_{target} values are in terms of the model's F values, which were calculated without using N:

> *If we were calculating with some abitrary F_{target}, we would have to consider N, but because our F_{target} is supposed to relate to a model which ignored N, we can omit N from our calculations.*

$$F = \frac{m + 0.005b}{N \cdot p}$$

Remember, we used a gradient to estimate the mass of a battery from it's capacity, so we'll refactor the

$$F = \frac{a \cdot c + m + 0.005b}{N \cdot p}$$

Where a is the estimated gradient and m is the base mass of the drone

$$F = \frac{a \cdot c + m + 0.005b}{N \cdot p}$$

$$N \cdot p \cdot F = a \cdot c + m + 0.005b$$

$$N \cdot p \cdot F - m - 0.005b = a \cdot c$$

$$c = \frac{N \cdot p \cdot F - m - 0.005b}{a}$$

As you can see, finding the optimal battery actually ends up having nothing to do with t *(however we could set a t value, then reverse the equation to find which values the variables would need to hold to achive this time)*. F is the main part of finding the battery, t is just how we score our results and compare performance. This also further amplifies, the importance of the gradient a, since if this is innacurate, our F and resultant c values will also be wrong. Another interesting finding from this is that the numerator of this expression is the mass of the target battery, since $c \cdot a = m$ so $c = \frac{m}{a}$, which is the form of our c formula.

This is the way to optimse how any general battery will perform against a given mode, but to improve the performance of the model itself, we tweak I_c, P_{MAX}, I_{MAX}, V; aiming to reduce the values in our t expressions. Carefully tuning these values within the constraints of the drone will maximise the flight time model, producing the best results for all batteries used.

ROADMAP

- [ ] Make command loading faster. Running `lithium --help` takes ages and running `lithium [command]` takes even longer.
- [ ] Make the progress bar in `lithium estimate` more accurate.
- [ ] Create an optimiser to recommend parts (or specs of parts) that will improve the model, while still being feasible (for example it won't just recommend setting mass to 0 even though this would drastically improve the model).