`Mortgage-Calculator` Mathematical Appendix

Matthew Kulec

Version 1.0: June 17, 2022

Deriving the Monthly Payment Formula

Let L_0 be the size of the loan, M be the monthly payment, and n be the duration of the mortgage in months. Furthermore, let r be the annual interest rate divided by 12 (resulting in a monthly interest rate). Note that $12r$ is not the quoted APR. APRs essentially package various one-time closing fees on top of the interest rate to make the comparison between different lenders feasible. However, putting aside one-time fees, the actual interest rate is what plays a crucial role in the calculation.

Every monthly payment is composed of interest and principal. The principal is what chips away at the loan, and the interest rate impedes one's ability to pay the loan faster (but of course you could always pay more every month, or take a smaller loan, for instance). At the end of the first month, the principal is the monthly payment minus the interest payed for that first month, which is $M - L_0 \cdot r$. Starting at the second month, the remaining balance is:

$$L_1 = L_0 - (M - L_0 \cdot r) = L_0 \cdot (1 + r) - M$$

At the end of the second month, the principal will now be the monthly payment minus the interest rate accumulated on whatever is left (which is L_1), leaving us with:

$$M - L_1 \cdot r$$

L_1 is smaller than L_0, and L_2 will certainly be smaller than L_1, etc, meaning that, at every monthly payment, more goes towards the reduction of the loan rather than the interest, and you (rather than the lender) own more and more of what you're trying to own. The size of the loan after two months shrinks down to:

$$L_2 = L_1 - (M - L_1 \cdot r) = L_1 \cdot (1 + r) - M = L_0 \cdot (1 + r)^2 - M \cdot (1 + r) - M$$

One can repeat the same exercise for the third month, but inductively, it can be shown that for the n^{th} month, the following formula holds:

$$L_n = L_0 \cdot (1 + r)^n - M \sum_{i=0}^{n-1} (1 + r)^i \tag{1}$$

Using the sum of geometric series, and that at the end of the n^{th} month, we want the loan to be payed off, the left hand size is zero and we are left with:

$$0 = L_0 \cdot (1 + r)^n - M \left(\frac{1 - (1 + r)^n}{1 - (1 + r)} \right) = L_0 \cdot (1 + r)^n - M \frac{(1 + r)^n - 1}{r}$$

Rearranging and solving for M, we arrive at the monthly payment M needed to pay off a loan of size L_0 in n months at a monthly interest rate of r:

$$\boxed{M = \frac{L_0 \cdot r \cdot (1 + r)^n}{(1 + r)^n - 1}}$$

Solving For L_0 and n

Solving for L_0 is as simple as writing it down:

$$\boxed{L_0 = M\frac{(1+r)^n - 1}{r \cdot (1+r)^n}}$$

Here's the algebra to solve for n:

$$M \cdot (1+r)^n - M = L_0 \cdot r \cdot (1+r)^n$$
$$(1+r)^n = \frac{M}{M - L_0 \cdot r}$$
$$n \cdot \log(1+r) = -\log\left(1 - \frac{L_0 \cdot r}{M}\right)$$

$$\boxed{n = \frac{-\log\left(1 - \frac{L_0 \cdot r}{M}\right)}{\log(1+r)}}$$

Solving For r

Now here's where things get interesting. The above equations can be expressed as a polynomial in terms of r. In general, there is no general algebraic formula for r if n is greater than 4, and it is more natural to resort to numerical root solving techniques like Newton's method or bisection method.

To that end, we expand $(1+r)^n$ to find the coefficients of the powers of r by applying the binomial theorem. These coefficients can then be efficiently passed as an array into `numpy`'s polynomial class `poly1d`, whose instance we can then use to perform function evaluations of that polynomial, without having to type the entire polynomial ourselves.

$$
\begin{gathered}
(M - L_0 \cdot r)(1+r)^n - M = 0 \\
(M - L_0 \cdot r)\sum_{k=0}^{n}\binom{n}{k} r^k - M = 0 \\
\left[\sum_{k=0}^{n} M\binom{n}{k} r^k - \sum_{k=0}^{n} L_0\binom{n}{k} r^{k+1}\right] - M = 0 \\
\left[M\binom{n}{0} + \sum_{k=1}^{n} M\binom{n}{k} r^k - \sum_{k=0}^{n-1} L_0\binom{n}{k} r^{k+1} - L_0\binom{n}{n} r^{n+1}\right] - M = 0 \\
\left[\cancel{M} + \sum_{k=1}^{n}\left\{M\binom{n}{k} - L_0\binom{n}{k-1}\right\} r^k - L_0 r^{n+1}\right] - \cancel{M} = 0
\end{gathered}
$$

Factoring out r, we aim to find r such that:

$$
\boxed{0 = f(r) = \sum_{k=1}^{n}\left\{M\binom{n}{k} - L_0\binom{n}{k-1}\right\} r^{k-1} - L_0 r^n}
$$

The derivative of $f(r)$ is

$$
f'(r) = \sum_{k=2}^{n}\left\{(k-1)M\binom{n}{k} - L_0 \cdot (k-1)\binom{n}{k-1}\right\} r^{k-2} - nL_0 r^{n-1}
$$

A numpy array of the coefficients (in brackets) can be constructed using `scipy.special.comb` and `numpy`'s broadcasting (vectorized operations).

Breaking down the k^{th} Monthly Payment

The amount of interest payed on the k^{th} month is just $L_{k-1}r$. Using formula 1 above,

$$
I_k = L_{k-1}r = L_0 r(1+r)^{k-1} - M\frac{(1+r)^{k-1} - 1}{1} = (L_0 r - M)(1+r)^{k-1} + M
$$

The principal for month k is whatever is left over from M:

$$P_k = M - I_k = (M - L_0 r)(1 + r)^{k-1}$$

Using these equations, we can develop the amortization schedule, which is a table that details the payments per month over the entire course of the mortgage.