Let the following definitions be made:

* the random variable denoting the value in today’s dollars of a HEP acquired today and either payed out (if the house sells before time ) or sold at time .
* the r.v. denoting the value in time T’s dollars of a HEP acquired at time T and either payed out or sold at time T + T1.
* the risk-free interest rate
* the p.d.f. of the distribution of the waiting time before a homeowner sells a home; assumed to be the exponential distribution
* the r.v. denoting the payout of a HEP claim that expires at time .
* the expectation operator.

The present value of the expectation of V(T) is just the discounted expectation of the payout multiplied by the probability that the HEP expires before time T, plus the probability that it doesn’t expire times the discounted expectation of the sale price at time T. We have

The expectation of is E[pS(t)-pM] = pS(0)exp() – pM, (where S is the price of the house and M is the initial mortgage amount) since the expected appreciation of a home in the neighborhood is the same as the expected appreciation of the neighborhood

Taken together, then, these integrals are easy. But the sale price of the HEP at time T is whatever anybody will pay for it, so we assume that the HEP sells for the maximum of its expected values to investors who intend to hold it for all possible time periods.

I don’t know how to approach the max part except by iterated numerical approximation, which I shall now attempt.

…

Okay, additional complication:

If the neighborhood μ is bigger than λ + *r* (i.e., if the average neighborhood appreciation is bigger than about 8.6% annually), then diverges, and there is no max{E[ V(T+T1)]}.

We no longer have to calculate the max, which is handy, but this throws my investor valuation model out of whack.

Probably the best thing is to calculate the variances, discount V by the investors’ risk aversion functions, and recalculate *r* if necessary: for if a HEP offers a risk-adjusted return better than *r*, then in fact *r* should be higher.

hile this handily avoids the problem of calculating the max