When I develop crossovers, I use a processor that serves as an active crossover configured with a Spice model. It allows me to change "components" using a configuration file. Once I have it fully optimized using Spice models, I build a physical crossover using passive components and verify that with measurements. But this is a development system, not one designed to be used at runtime.

The biggest problem you'll have using off-the-shelf active crossovers is you have to develop the filters. Some don't have the flexibility to do much more than make a basic "textbook" filter, e.g. Butterworth, Linkwitz-Riley, etc. But the filters in my speakers don't fit into any of those molds, so crossovers like that give unsatisfactory results. Others can be programmed, but then the user has to provide the algorithm to obtain the proper transfer function, and that's not trivial. So most people resort to basic filters, even with the programmable crossovers, and like I said above, that approach doesn't work well.

So the bottom line is, you need to be able to match the transfer function of the passive crossover. If you cannot do this, then the passive crossover will be better than the active crossover. You will negate the benefits of going active by having filters that don't work right for the loudspeaker, and the on-axis and off-axis response will suffer.

Still, I don't mean to discourage you. If you want to pursue it, here is a link that will show you what you need to do:
Crossover optimization for DI-matched two-way speakers