INTERACTIVE PARALLELISM
def qualityup(tasks=0, precflag='d'):
    from phcpy.families import cyclic
    from phcpy.solver import solve
    c7 = cyclic(7)
    tstart = perf_counter()
    s = solve(c7, verbose=False, tasks=tstart,
              precflag=precflag, checkin=False)
    return perf_counter() - tstart

The blackbox solver’s elapsed performance
in double, double double, and quad double precision:

<table>
<thead>
<tr>
<th>precision</th>
<th>d</th>
<th>dd</th>
<th>qd</th>
<th>tasks</th>
<th>8</th>
<th>16</th>
<th>32</th>
</tr>
</thead>
<tbody>
<tr>
<td>Elapsed performance</td>
<td>5.45</td>
<td>42.41</td>
<td>604.91</td>
<td>dd</td>
<td>7.56</td>
<td>5.07</td>
<td>3.88</td>
</tr>
<tr>
<td>Overhead factor</td>
<td>1.00</td>
<td>7.41</td>
<td>110.99</td>
<td>qd</td>
<td>96.08</td>
<td>65.82</td>
<td>44.35</td>
</tr>
</tbody>
</table>

The overhead of extra precision is compensated by multithreading.

Interpolated elapsed performances for quad double arithmetic:

We can numerically solve polynomial systems in phcpy, using fast and reliable homotopy continuation methods.

SOLUTION SETS
A numerical irreducible decomposition includes representations for all positive dimensional solution sets.

Consider two equations defining the twisted cubic:

```
pols = ['x*y - z', 'x^2 - y']
```

(1) A witness set provides generic points:
```
from phcpy.ssets import embed
from phcpy.solver import solve
emb = embed(3, 1, pols)
sols = solve(emb, verbose=False)
print('# generic points:', len(sols))
```

Three points lie at the intersection of this cubic with a random plane.

(2) A series expansion develops from some point(s) in a coordinate hyperplane. For the twisted cubic, the series is exact after the first term.
```
from phcpy.maps import solve_binomials
maps = solve_binomials(3, pols, puretopdim=True)
for sol in maps: print(sol)
```

The solution gives \((t,t^2,P)\), the parametric representation of the twisted cubic.