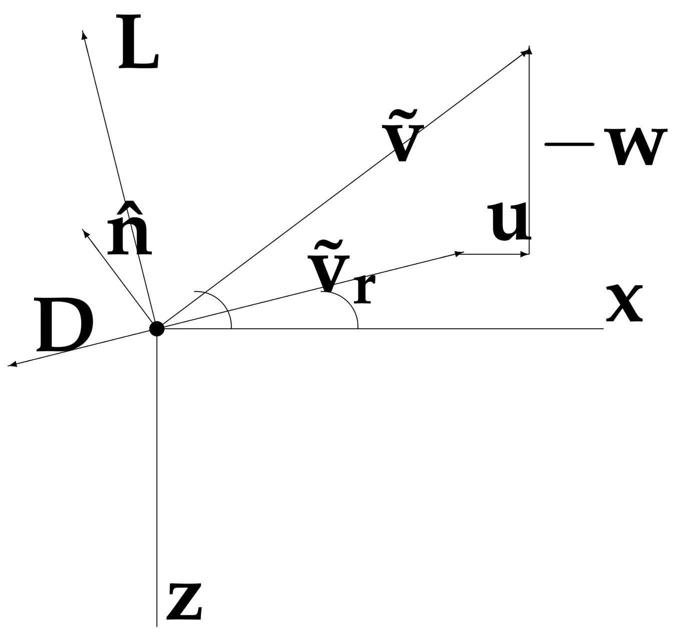
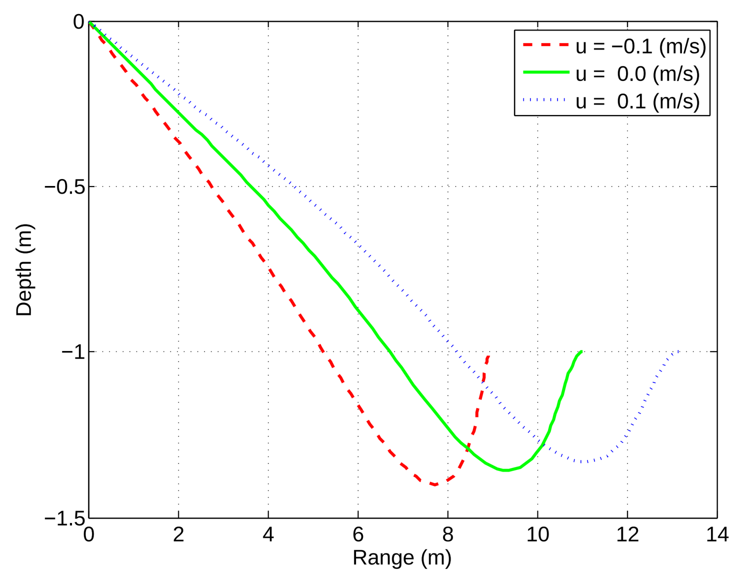
Parallel Matlab: `fmincon`

Max Range Undersea Glide

VirginiaTech Invent the Future John V. Burkardt & Eugene M. Cliff ICAM

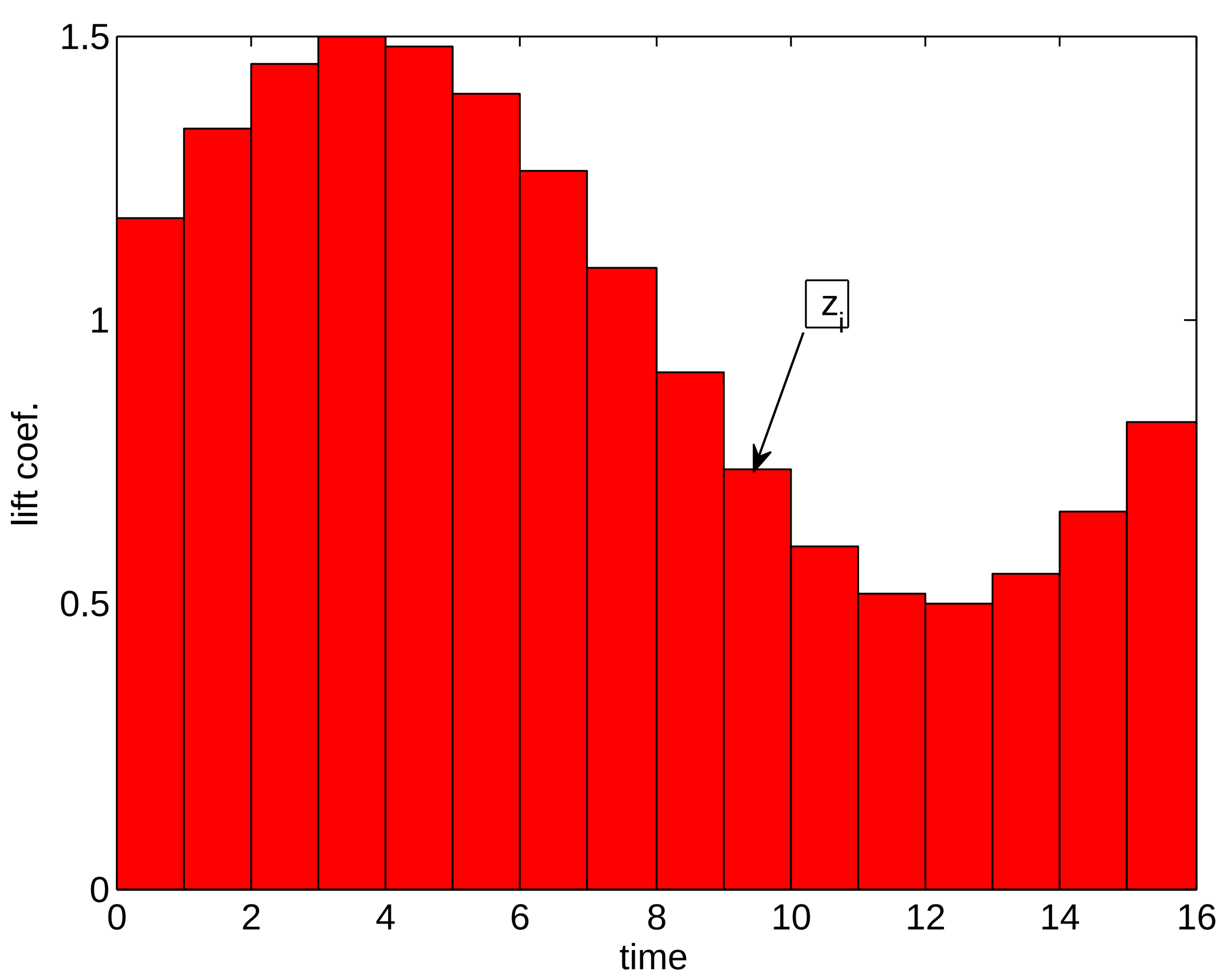
Mechanics





NLP Problem

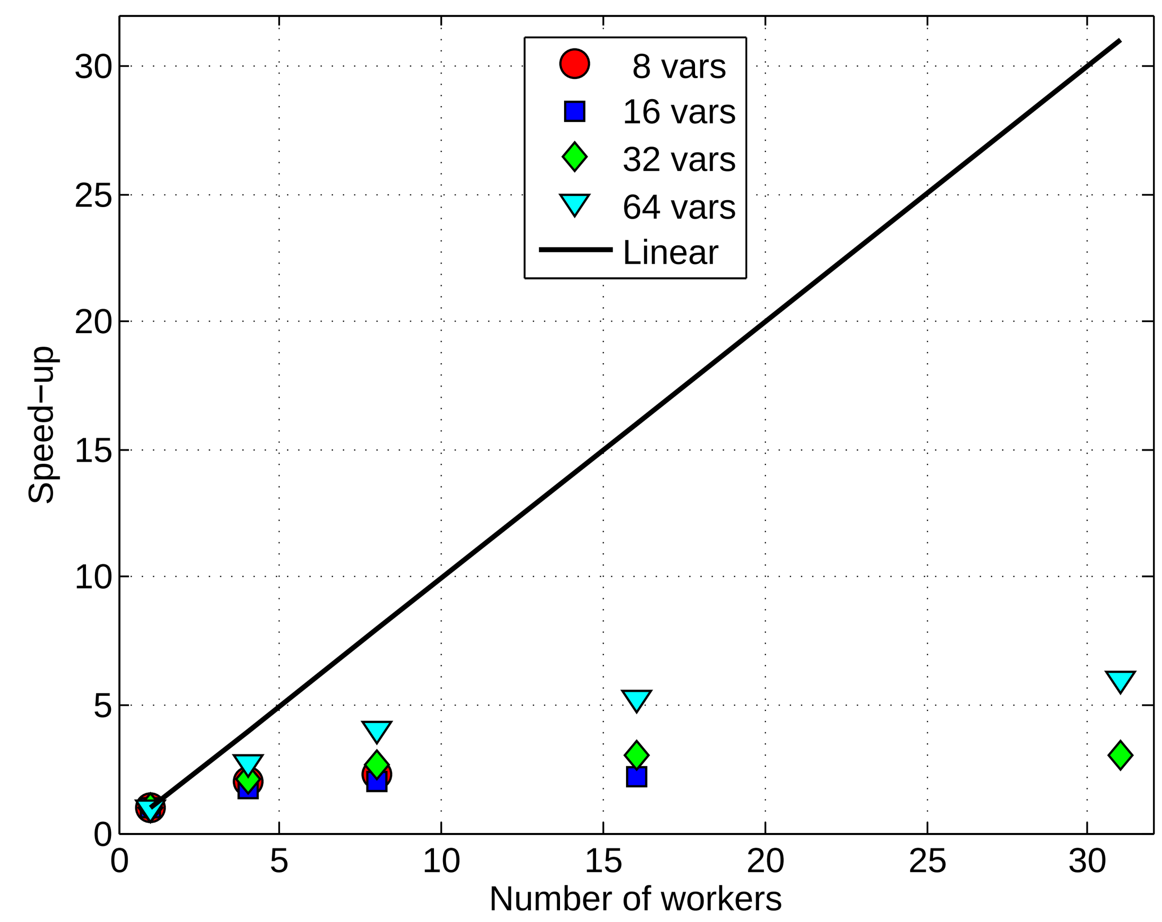
- State variables V, γ, x, z
- Control variable C_L
- Parameters $z_i \sim C_L(t_i)$
- Objective/Contraint $\Longrightarrow$ solve I-V-P



Code Fragment

```
%
OPTIONS = optimset(OPTIONS,
     ....'UseParallel','always');
hndl_obj =@(z) obj2_post(z, DATA);
hndl_con=@(z) con_post(z, DATA);
%
matlabpool open 4
save_time( 1, : ) = clock;
Z_star= fmincon(...
hndl_obj,Z0,[],[],[],[],LB,UB,...
hndl_con, OPTIONS);
save_time( 2, : ) = clock;
execute= ...
disp(['execute=' num2str(execute)])
matlabpool close
```

Timing Results



n labs	time (s) 8 vars	time (s) 16 vars	time (s) 32 vars	time (s) 64 vars
0	60	123.6	246	695
4	29.7	70.9	118	254
8	26.7	61.4	92	173
16	na	57	82	131
31	na	na	82	115

Discussion

A modest number of processors provide some speed-up, but the improvement falls off quickly with the number of processors. This is in large part due to the algorithm. We are using parallel processing only in the calculation of finite-difference estimates for Jacobians. The optimization algorithm requires additional function evaluations as part of its step-size selection strategy. In the problem with 32 variables on 16 processors the code ran a total of 21 iterations using 846 function evaluations. 672 evaluations were for finite-difference estimates, and the remaining 174 were run in the step-size selection procedure (on a single processor). For each gradient calculation, each worker would ideally perform 2 function evaluations for a total of 42 function cycles. Add to this the 174 function cycles required in the step-size selection procedure for a total of 216 function cycles - the ideal speed-up would be $\frac{846}{216} \approx 3.9$. The achieved speed up was $\frac{266}{82} = 3$.