FROM PROBLEM TO SOLUTION IN ONE CL1CK

Prof. Paolo Bientinesi

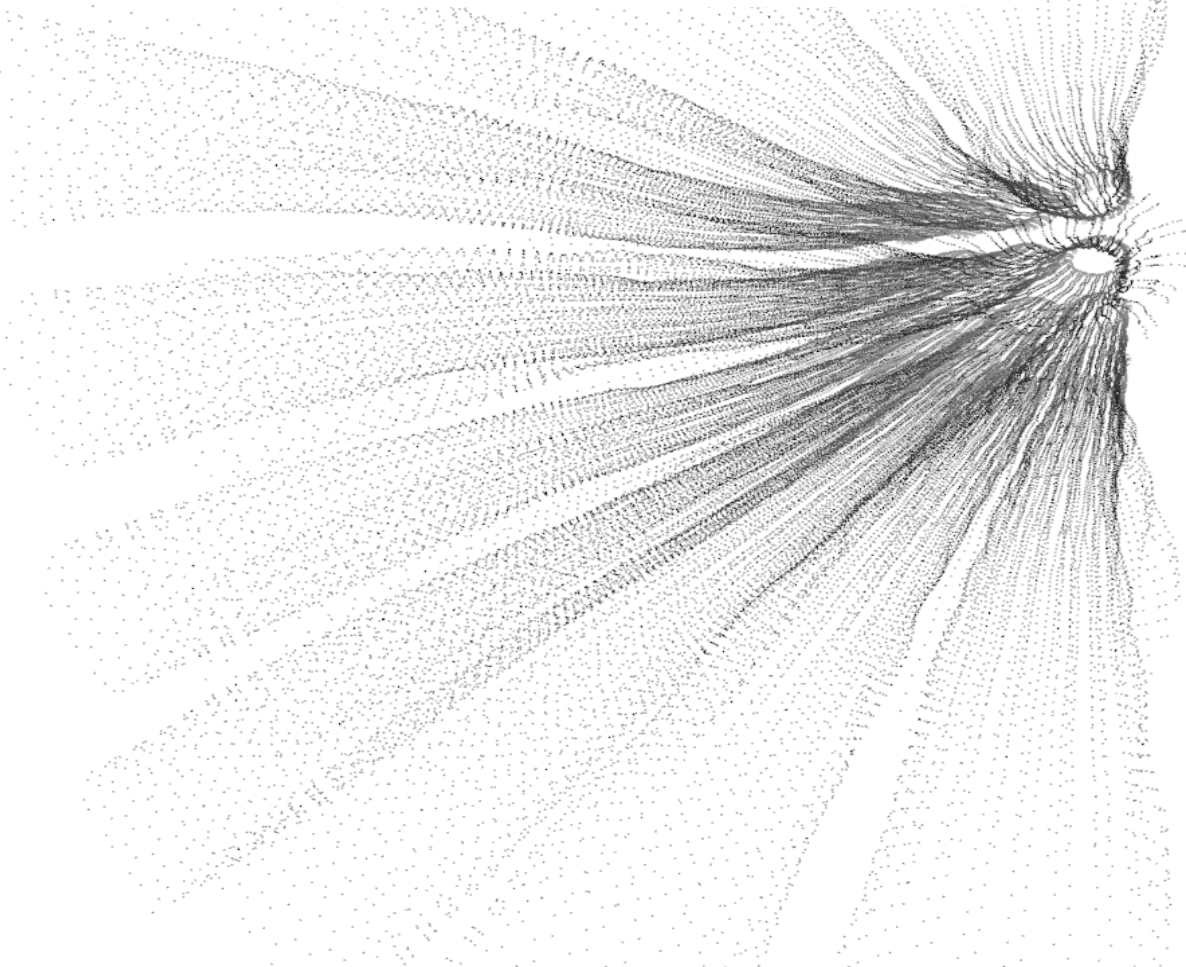
pauldj@cs.umu.se

UMEÅ UNIVERSITY

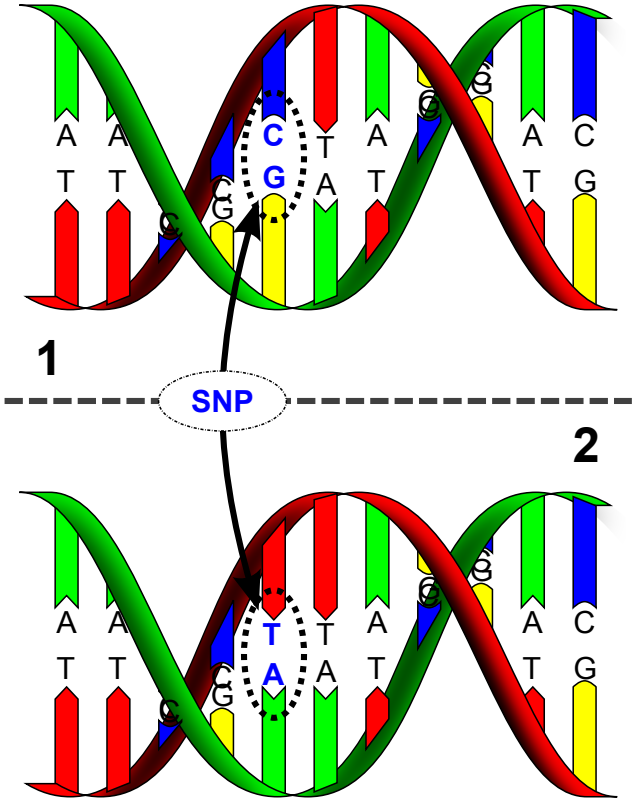
The World of Computing

UMEÅ UNIVERSITY

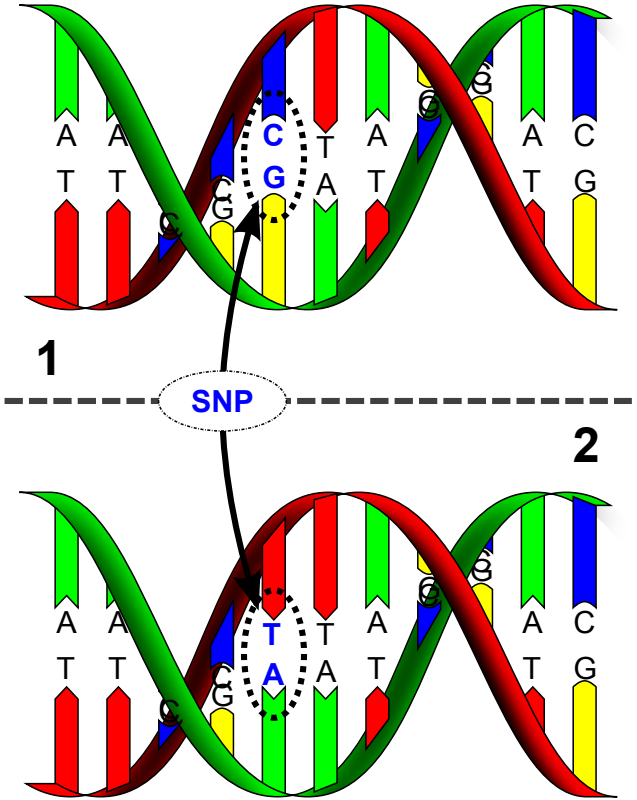
The World of Computing

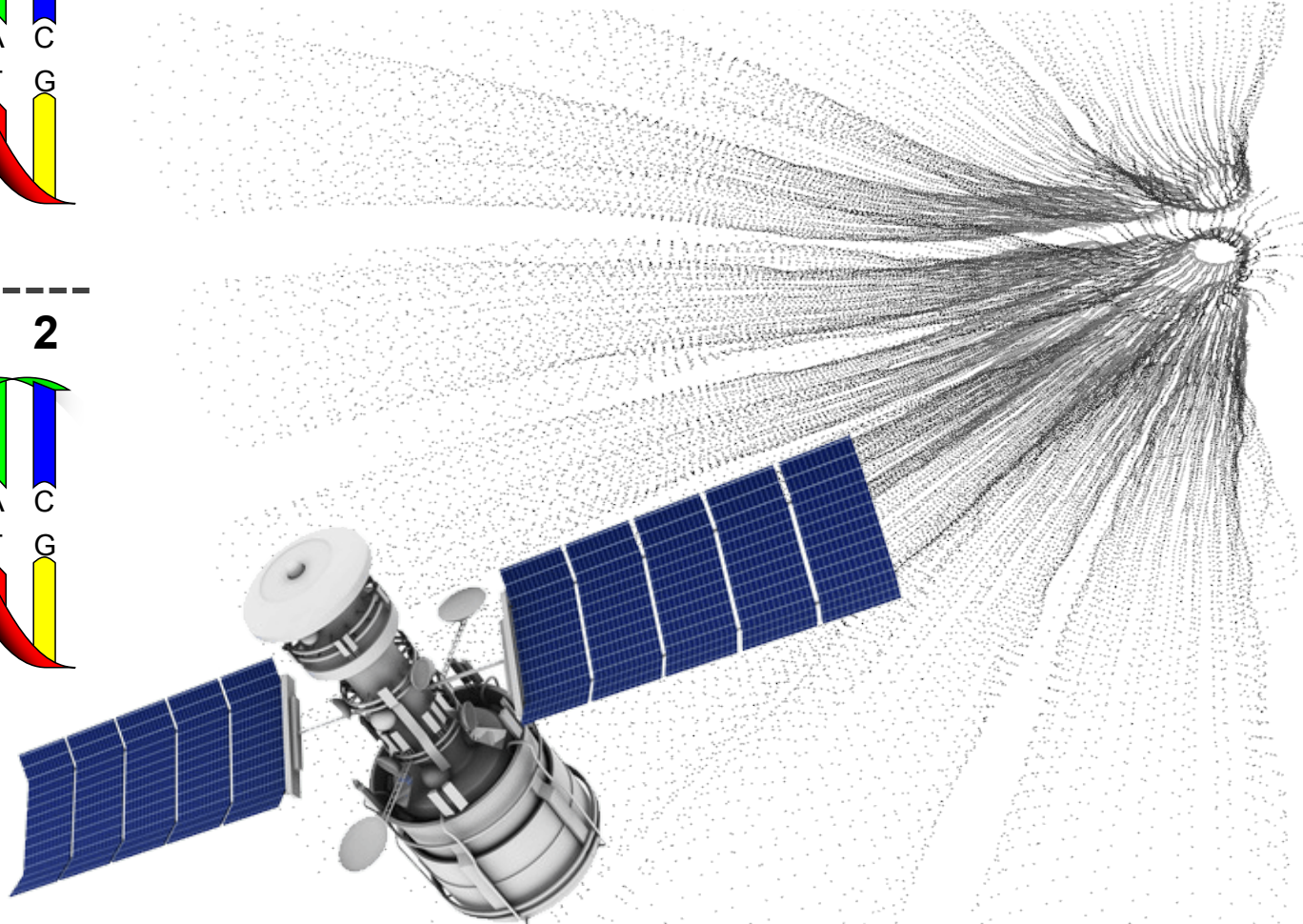


The World of Computing

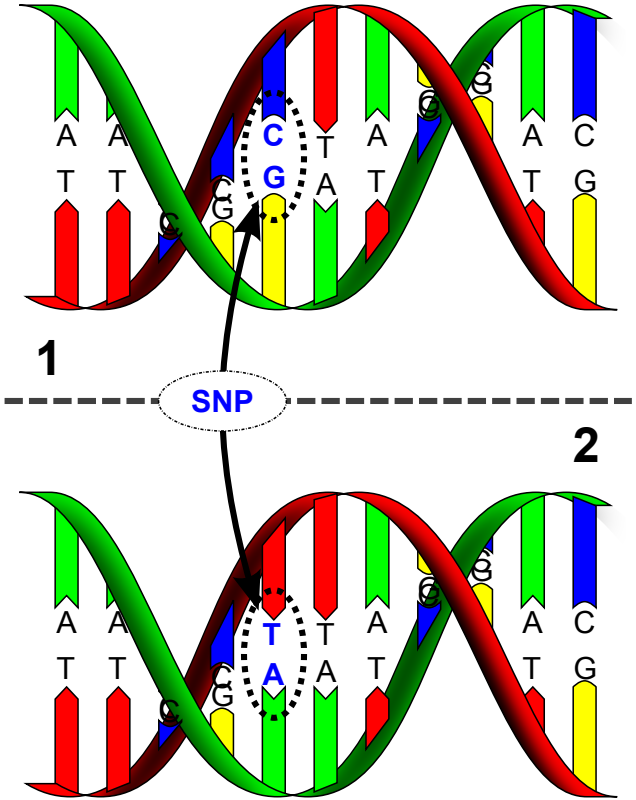


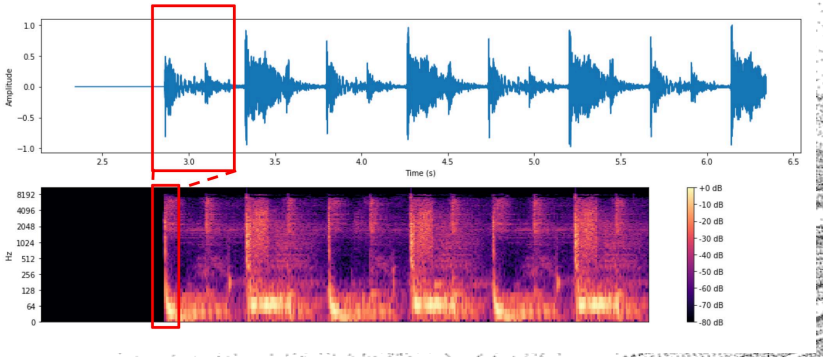
The World of Computing

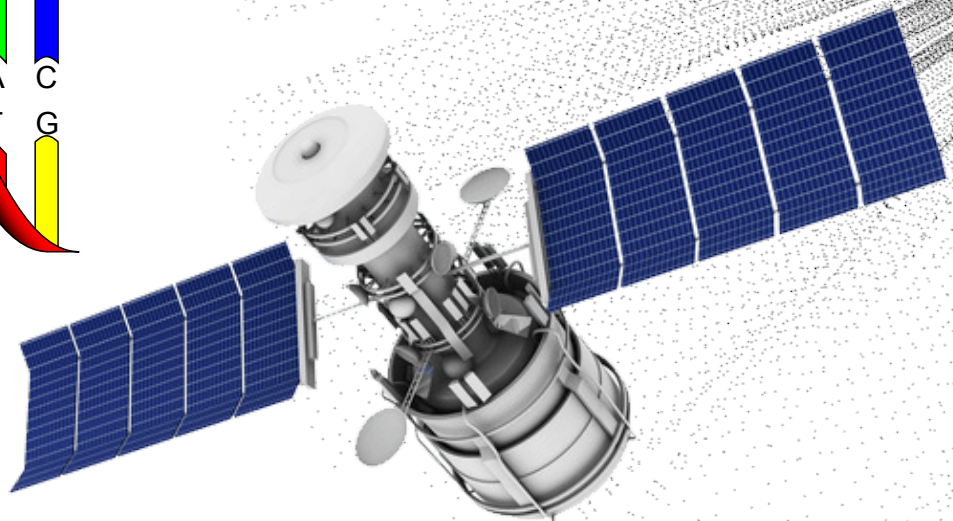




The World of Computing

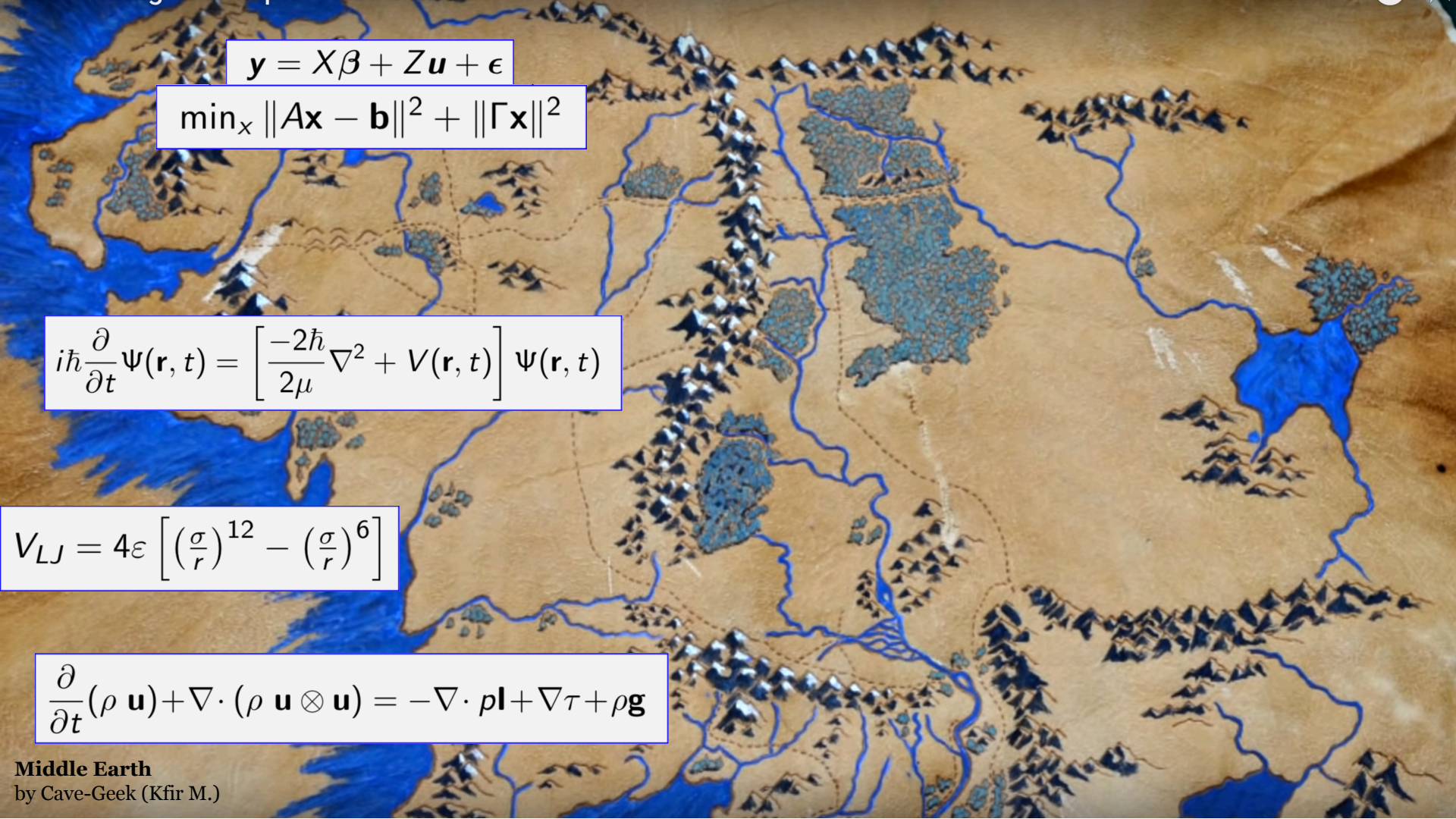






Countries

Middle Earth
by Cave-Geek (Kfir M.)



$$\boldsymbol{y} = X\boldsymbol{\beta} + Z\boldsymbol{u} + \boldsymbol{\epsilon}$$

$$\min_x \|A\mathbf{x} - \mathbf{b}\|^2 + \|\Gamma\mathbf{x}\|^2$$

$$i\hbar\frac{\partial}{\partial t}\Psi(\mathbf{r},t) = \left[\frac{-2\hbar}{2\mu}\nabla^2 + V(\mathbf{r},t)\right]\Psi(\mathbf{r},t)$$

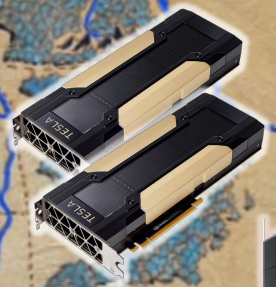
$$V_{LJ} = 4\varepsilon\left[\left(\frac{\sigma}{r}\right)^{12} - \left(\frac{\sigma}{r}\right)^{6}\right]$$

$$\frac{\partial}{\partial t}(\rho\ \mathbf{u}) + \nabla\cdot(\rho\ \mathbf{u}\otimes\mathbf{u}) = -\nabla\cdot p\mathbf{I} + \nabla\tau + \rho\mathbf{g}$$

Middle Earth
by Cave-Geek (Kfir M.)

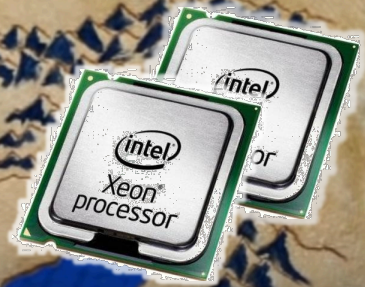
$$\boldsymbol{y} = X\boldsymbol{\beta} + Z\boldsymbol{u} + \boldsymbol{\epsilon}$$

$$\min_x \|A\mathbf{x} - \mathbf{b}\|^2 + \|\Gamma\mathbf{x}\|^2$$



$$i\hbar\frac{\partial}{\partial t}\Psi(\mathbf{r},t) = \left[\frac{-2\hbar}{2\mu}\nabla^2 + V(\mathbf{r},t)\right]\Psi(\mathbf{r},t)$$

$$V_{LJ} = 4\varepsilon\left[\left(\frac{\sigma}{r}\right)^{12} - \left(\frac{\sigma}{r}\right)^{6}\right]$$



$$\frac{\partial}{\partial t}(\rho\ \mathbf{u}) + \nabla\cdot(\rho\ \mathbf{u}\otimes\mathbf{u}) = -\nabla\cdot p\mathbf{I} + \nabla\tau + \rho\mathbf{g}$$

Middle Earth
by Cave-Geek (Kfir M.)

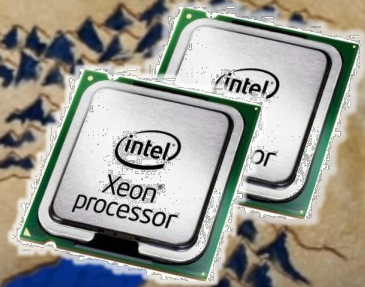
$$\boldsymbol{y} = X\boldsymbol{\beta} + Z\boldsymbol{u} + \boldsymbol{\epsilon}$$

$$\min_x \|A\mathbf{x} - \mathbf{b}\|^2 + \|\Gamma\mathbf{x}\|^2$$

$$i\hbar\frac{\partial}{\partial t}\Psi(\mathbf{r},t) = \left[\frac{-2\hbar}{2\mu}\nabla^2 + V(\mathbf{r},t)\right]\Psi(\mathbf{r},t)$$

$$V_{LJ} = 4\varepsilon\left[\left(\frac{\sigma}{r}\right)^{12} - \left(\frac{\sigma}{r}\right)^{6}\right]$$

$$\frac{\partial}{\partial t}(\rho\,\mathbf{u}) + \nabla\cdot(\rho\,\mathbf{u}\otimes\mathbf{u}) = -\nabla\cdot p\mathbf{I} + \nabla\tau + \rho\mathbf{g}$$



Middle Earth
by Cave-Geek (Kfir M.)

$$y = X\beta + Zu + \epsilon$$

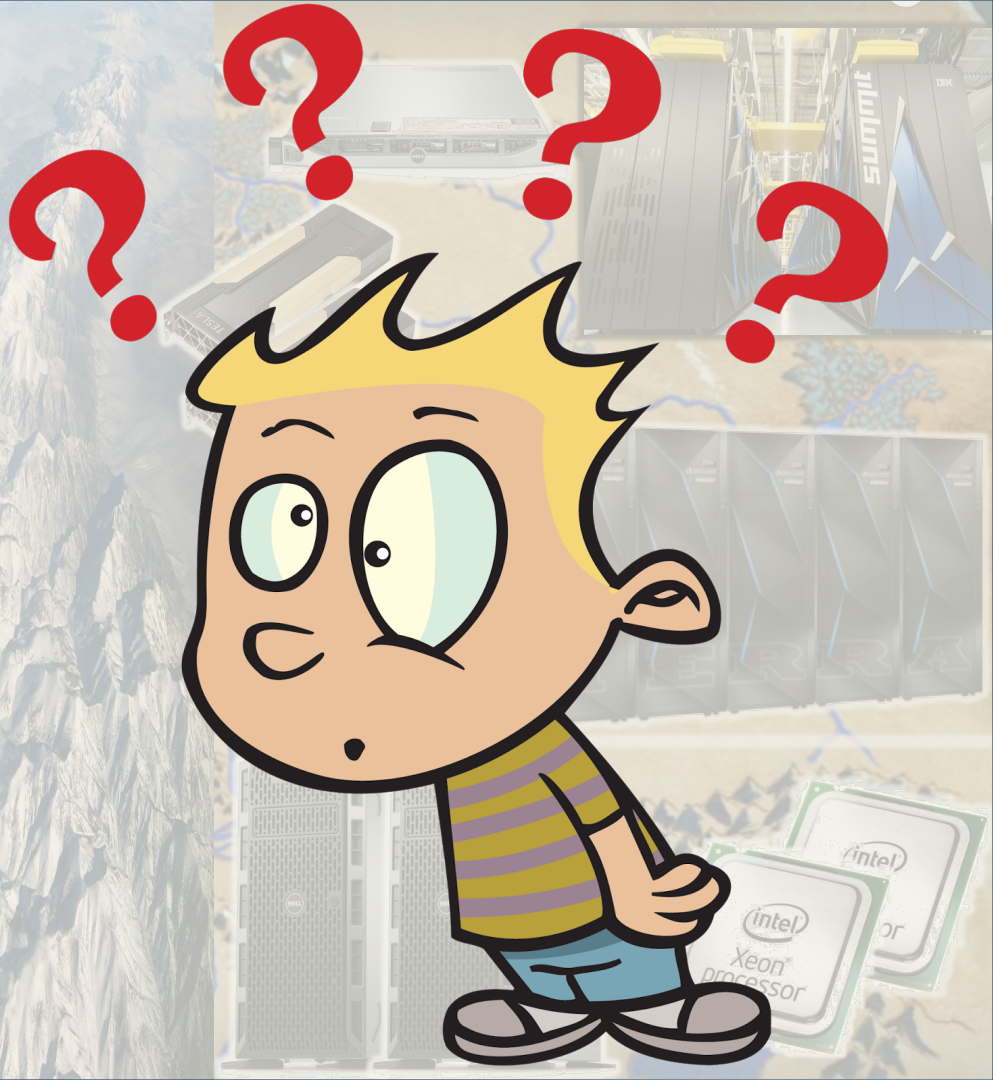
$$\min_x \|Ax - b\|^2 + \|\Gamma x\|^2$$

$$i\hbar\frac{\partial}{\partial t}\Psi(\mathbf{r},t) = \left[\frac{-2\hbar}{2\mu}\nabla^2 + V(\mathbf{r},t)\right]\Psi(\mathbf{r},t)$$

$$V_{LJ} = 4\varepsilon\left[\left(\frac{\sigma}{r}\right)^{12} - \left(\frac{\sigma}{r}\right)^{6}\right]$$

$$\frac{\partial}{\partial t}(\rho\,\mathbf{u}) + \nabla\cdot(\rho\,\mathbf{u}\otimes\mathbf{u}) = -\nabla\cdot p\mathbf{I} + \nabla\tau + \rho\mathbf{g}$$

Middle Earth
by Cave-Geek (Kfir M.)



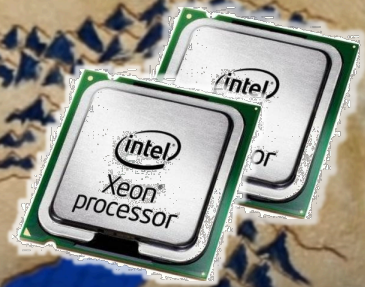
$$\boldsymbol{y} = X\boldsymbol{\beta} + Z\boldsymbol{u} + \boldsymbol{\epsilon}$$

$$\min_x \|A\mathbf{x} - \mathbf{b}\|^2 + \|\Gamma\mathbf{x}\|^2$$

$$i\hbar\frac{\partial}{\partial t}\Psi(\mathbf{r},t) = \left[\frac{-2\hbar}{2\mu}\nabla^2 + V(\mathbf{r},t)\right]\Psi(\mathbf{r},t)$$

$$V_{LJ} = 4\varepsilon\left[\left(\frac{\sigma}{r}\right)^{12} - \left(\frac{\sigma}{r}\right)^{6}\right]$$

$$\frac{\partial}{\partial t}(\rho\,\mathbf{u}) + \nabla\cdot(\rho\,\mathbf{u}\otimes\mathbf{u}) = -\nabla\cdot p\mathbf{I} + \nabla\tau + \rho\mathbf{g}$$



Middle Earth
by Cave-Geek (Kfir M.)

$$y = X\beta + Zu + \epsilon$$

$$\frac{\partial}{\partial t}(\rho\,\mathbf{u}) + \nabla\cdot(\rho\,\mathbf{u}\otimes\mathbf{u}) = -\nabla\cdot p\mathbf{I} + \nabla\tau + \rho\mathbf{g}$$

Middle Earth
by Cave-Geek (Kfir M.)

Programmability

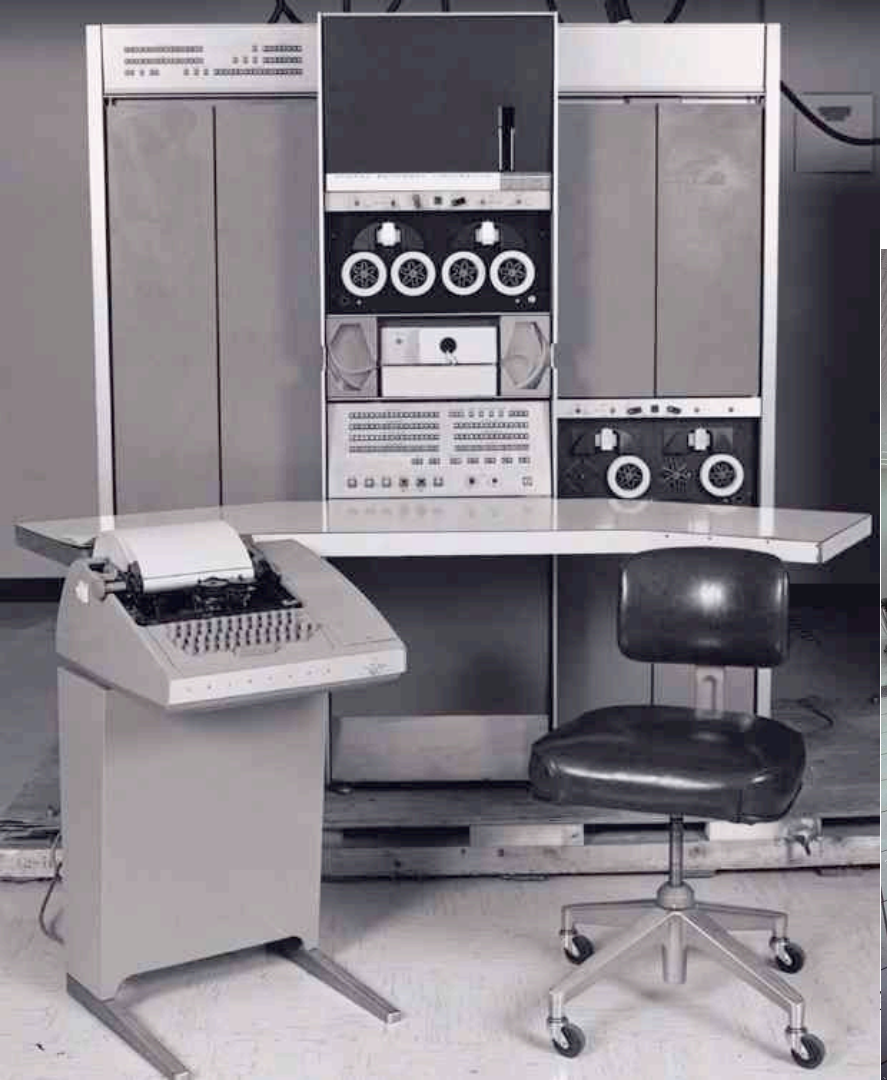
UMEÅ UNIVERSITY

Programmability

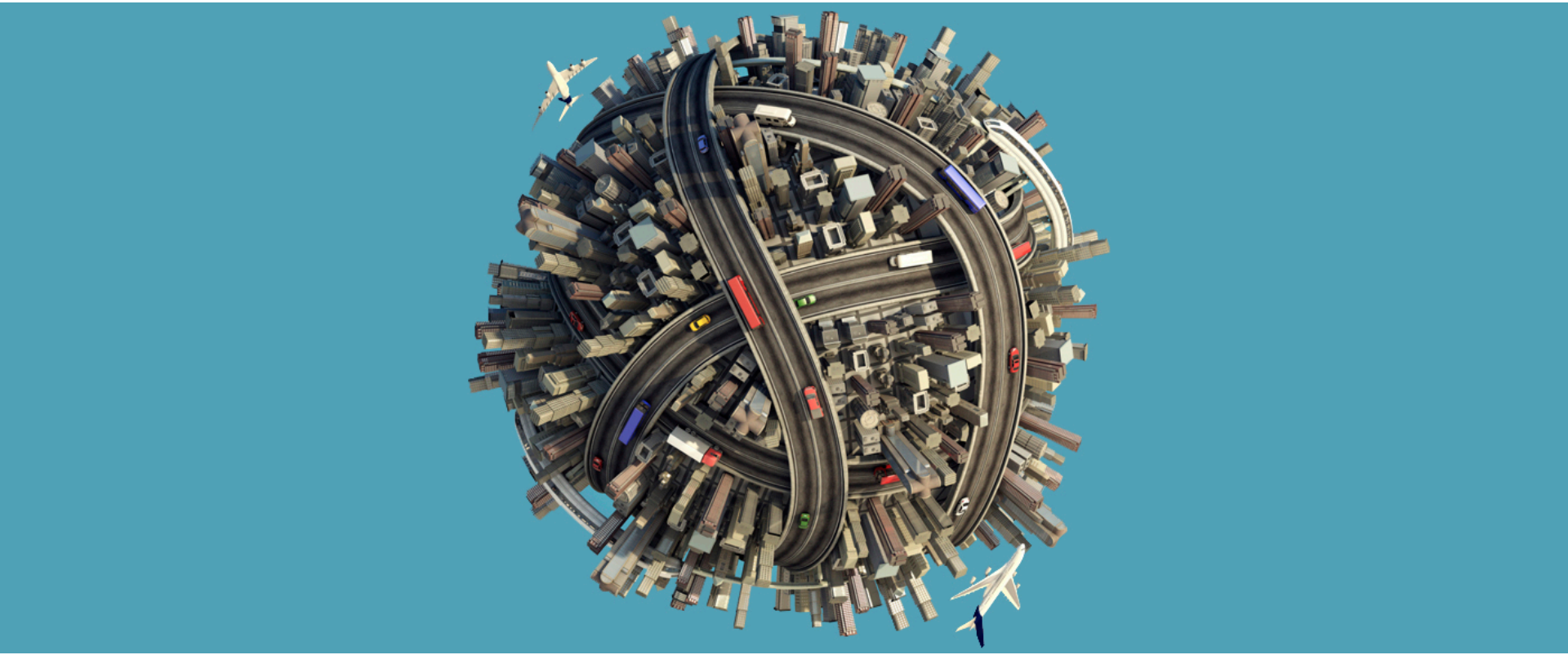
1960's
- computers very difficult to program
- programs easy to optimize

Since **1980**'s
- computers increasingly easy to program
- programs increasingly difficult to optimize

“Good old days”

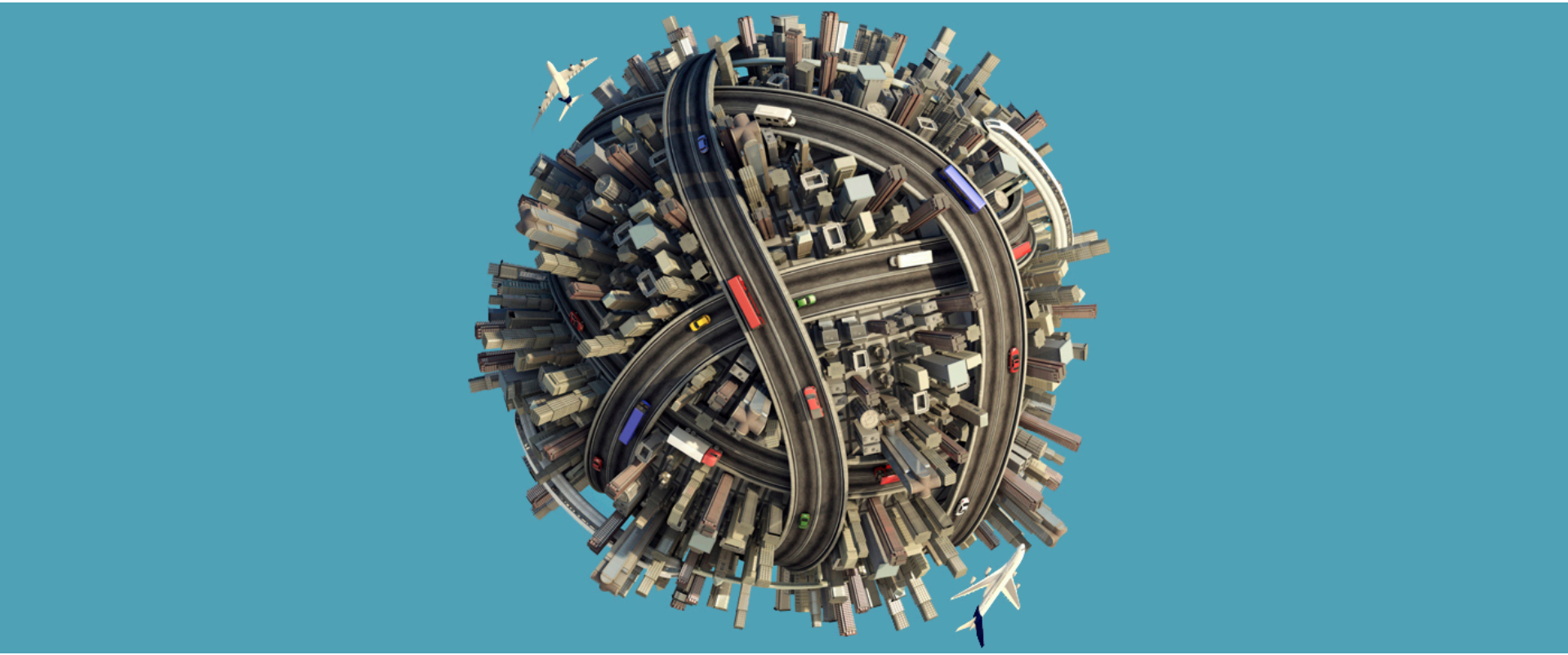


One country, one language



Cheong Suk-Wai, A borderless world

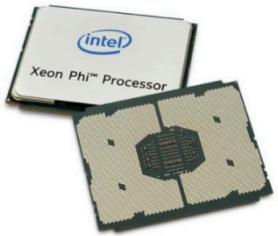
~~One country, one language~~

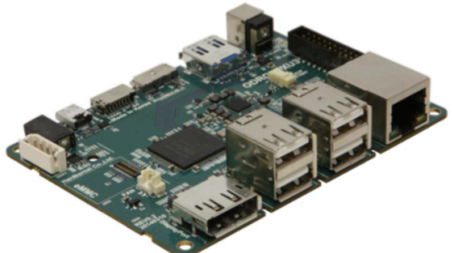


Cheong Suk-Wai, A borderless world

Language mismatch

Language mismatch





MUL ADD MOV

MOVAPD

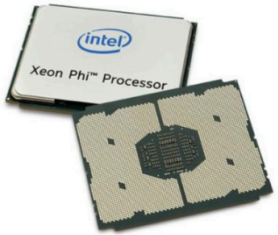
VFMADDPD ...

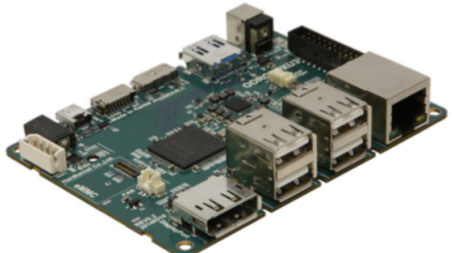
$$K_k := P_k^b H^T (H P_k^b H^T + R)^{-1};\ x_k^a := x_k^b + K_k(z_k - H x_k^b);\ P_k^a := (I - K_K H)\, P_k^b$$

$$\begin{aligned}
\tau_{ij}^{ab} &= t_{ij}^{ab} + \frac{1}{2} P_b^a P_j^i t_i^a t_j^b, \\
\tilde{F}_e^m &= f_e^m + \sum_{fn} v_{ef}^{mn} t_n^f, \\
\tilde{F}_e^a &= (1 - \delta_{ae}) f_e^a - \sum_m \tilde{F}_e^m t_m^a - \frac{1}{2} \sum_{mnf} v_{ef}^{mn} t_{mn}^{af} + \sum_{fn} v_{ef}^{an} t_n^f, \\
\tilde{F}_i^m &= (1 - \delta_{mi}) f_i^m + \sum_e \tilde{F}_e^m t_i^e + \frac{1}{2} \sum_{nef} v_{ef}^{mn} t_{in}^{ef} + \sum_{fn} v_{if}^{mn} t_n^f,
\end{aligned}$$

$$\Lambda := S(S^T A W A S)^{-1} S^T;\ \Theta := \Lambda A W;\ M_k := X_k A - I$$
$$X_{k+1} := X_k - M_k \Theta - (M_k \Theta)^T + \Theta^T (A X_k A - A) \Theta$$

$$\begin{cases} C_\dagger := P C P^T + Q \\ K := C_\dagger H^T (H C_\dagger H^T)^{-1} \end{cases}$$





MUL ADD MOV

MOVAPD

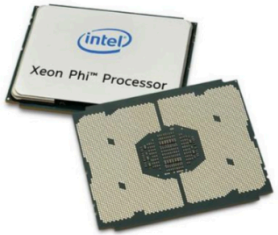
VFMADDPD

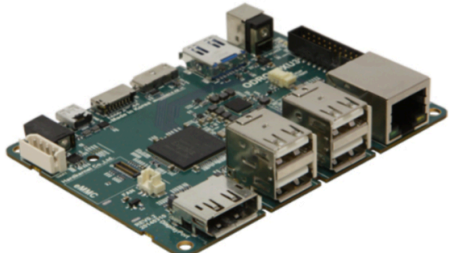
$$K_k := P_k^b H^T (H P_k^b H^T + R)^{-1};\ x_k^a := x_k^b + K_k(z_k - H x_k^b);\ P_k^a := (I - K_K H)\, P_k^b$$

$$\begin{aligned}
\tau_{ij}^{ab} &= t_{ij}^{ab} + \frac{1}{2} P_b^a P_j^i t_i^a t_j^b, \\
\tilde{F}_e^m &= f_e^m + \sum_{fn} v_{ef}^{mn} t_n^f, \\
\tilde{F}_e^a &= (1 - \delta_{ae}) f_e^a - \sum_m \tilde{F}_e^m t_m^a - \frac{1}{2} \sum_{mnf} v_{ef}^{mn} t_{mn}^{af} + \sum_{fn} v_{ef}^{an} t_n^f, \\
\tilde{F}_i^m &= (1 - \delta_{mi}) f_i^m + \sum_e \tilde{F}_e^m t_i^e + \frac{1}{2} \sum_{nef} v_{ef}^{mn} t_{in}^{ef} + \sum_{fn} v_{if}^{mn} t_n^f,
\end{aligned}$$

$$\Lambda := S(S^T A W A S)^{-1} S^T;\ \Theta := \Lambda A W;\ M_k := X_k A - I$$
$$X_{k+1} := X_k - M_k \Theta - (M_k \Theta)^T + \Theta^T (A X_k A - A) \Theta$$

$$\begin{cases} C_\dagger := P C P^T + Q \\ K := C_\dagger H^T (H C_\dagger H^T)^{-1} \end{cases}$$





MUL ADD MOV

MOVAPD

VFMADDPD ...

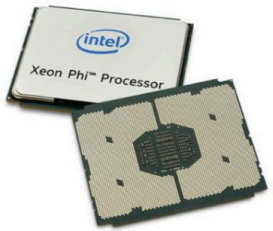
$$K_k := P_k^b H^T (H P_k^b H^T + R)^{-1};\ x_k^a := x_k^b + K_k(z_k - H x_k^b);\ P_k^a := (I - K_K H)\, P_k^b$$

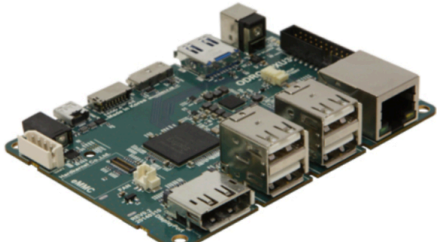
$$\begin{cases} C_\dagger := PCP^T + Q \\ K := C_\dagger H^T (H C_\dagger H^T)^{-1} \end{cases}$$

$$\Lambda := S(S^T AWAS)^{-1} S^T;\ \Theta := \Lambda A W;\ M_k := X_k A - I$$
$$X_{k+1} := X_k - M_k \Theta - (M_k \Theta)^T + \Theta^T (A X_k A - A)\Theta$$

$y := \alpha x + y$ | $LU = A$ | $\cdots$ | $C := \alpha AB + \beta C$

$X := A^{-1}B$ | $C := AB^T + BA^T + C$ | $X := L^{-1}ML^{-T}$ | $QR = A$





MUL | ADD | MOV

MOVAPD

VFMADDPD

$$K_k := P_k^b H^T (H P_k^b H^T + R)^{-1};\ x_k^a := x_k^b + K_k(z_k - H x_k^b);\ P_k^a := (I - K_K H)\, P_k^b$$

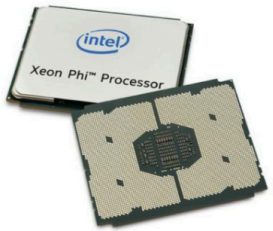
$$\begin{cases} C_\dagger := PCP^T + Q \\ K := C_\dagger H^T (H C_\dagger H^T)^{-1} \end{cases}$$

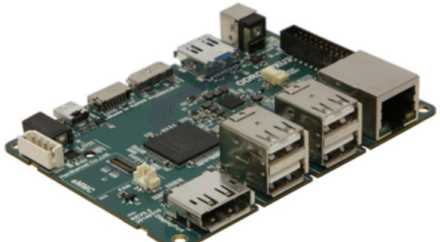
$$\Lambda := S(S^T AWAS)^{-1} S^T;\ \Theta := \Lambda A W;\ M_k := X_k A - I$$
$$X_{k+1} := X_k - M_k \Theta - (M_k \Theta)^T + \Theta^T (A X_k A - A)\Theta$$

?

$y := \alpha x + y$ | $LU = A$ | $\cdots$ | $C := \alpha AB + \beta C$

$X := A^{-1}B$ | $C := AB^T + BA^T + C$ | $X := L^{-1}ML^{-T}$ | $QR = A$





MUL | ADD | MOV

MOVAPD

VFMADDPD

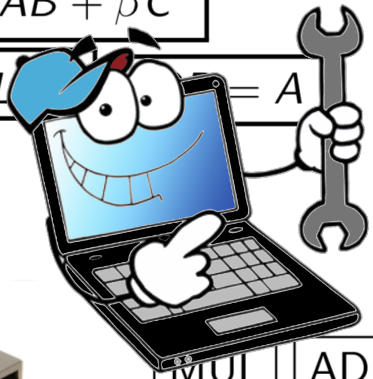
$$K_k := P_k^b H^T (H P_k^b H^T + R)^{-1};\ x_k^a := x_k^b + K_k(z_k - H x_k^b);\ P_k^a := (I - K_K H)\, P_k^b$$

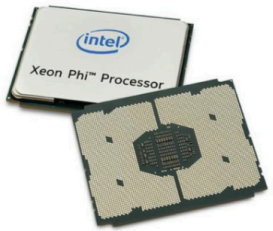
$$\begin{cases} C_\dagger := PCP^T + Q \\ K := C_\dagger H^T (H C_\dagger H^T)^{-1} \end{cases}$$

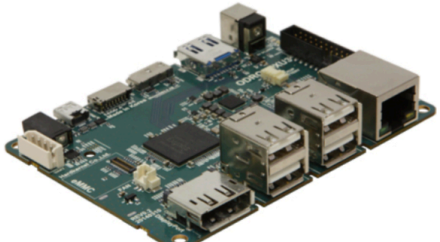
$$\Lambda := S(S^T AWAS)^{-1} S^T;\ \Theta := \Lambda A W;\ M_k := X_k A - I$$
$$X_{k+1} := X_k - M_k \Theta - (M_k \Theta)^T + \Theta^T (A X_k A - A)\Theta$$

$y := \alpha x + y$ | $LU = A$ | $\cdots$ | $C := \alpha AB + \beta C$

$X := A^{-1}B$ | $C := AB^T + BA^T + C$ | $X := L^{-1}ML$ | $= A$







MUL | ADD | MOV

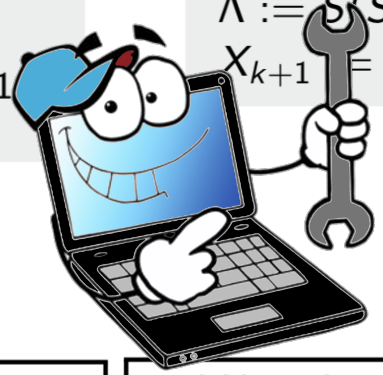
MOVAPD

VFMADDPD

$$K_k := P_k^b H^T (H P_k^b H^T + R)^{-1};\ x_k^a := x_k^b + K_k(z_k - H x_k^b);\ P_k^a := (I - K_K H)\, P_k^b$$

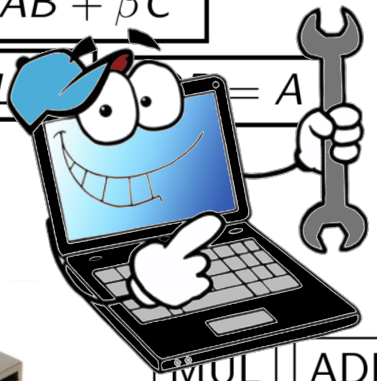
$$\begin{cases} C_\dagger := PCP^T + Q \\ K := C_\dagger H^T (H C_\dagger H^T)^{-1} \end{cases}$$

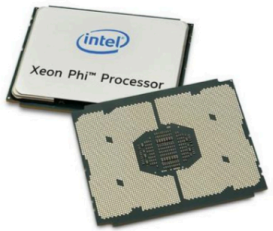
$$\Lambda := S(S^T AWAS)^{-1} S^T;\ \Theta := \Lambda AW;\ M_k := X_k A - I$$
$$X_{k+1} = X_k - M_k\Theta - (M_k\Theta)^T + \Theta^T (A X_k A - A)\Theta$$

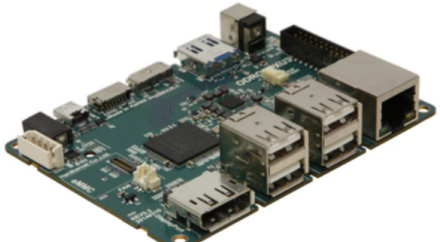


$y := \alpha x + y$ | $LU = A$ | $\cdots$ | $C := \alpha AB + \beta C$

$X := A^{-1}B$ | $C := AB^T + BA^T + C$ | $X := L^{-1}ML$ | $= A$







MUL | ADD | MOV

MOVAPD

VFMADDPD

PotC

EAM Potential

```
parameter cutoff = file(1);
parameter grid_size_rho = file(2);
parameter grid_size = file(3);
parameter rho(i : atom_type; j : atom_type; r :
   ↪ distance) =
    spline_grid(rho, 3, 0, grid_size, 3);
parameter F(i : atom_type; r : real) =
    spline_grid(F, 3, 0, grid_size_rho, 1);
parameter phi(i : atom_type; j : atom_type; r :
   ↪ distance) =
    spline_grid(phi, 3, 0, grid_size, 3);

peratom rho_sum(i : atom) = sum(j : neighbors(i,
   ↪ cutoff)) rho(i, j, r(i, j));

energy sum(i : all_atoms) F(i, rho_sum(i));
energy sum(i : all_atoms) sum(j : neighbors_half(i,
   ↪ cutoff)) phi(i, j, r(i, j)) / r(i, j);
```

PotC

1177
lines of code

```
bvec t_0 = bvec::full();
  for (int t_1 = iafrom; t_1 < iato; t_1 += fvec::VL){
   ivec i_i_2 = ivec::set_consecutive() + ivec::set1(t_1);
   bvec t_2 = bvec::kand(bvec::full(), ivec::cmplt(i_i_2, ivec::set1(iato)));
   fvec::mask_scatter_double_zero(peratom_rho_sum, t_2, i_i_2);
   fvec::mask_scatter_double_zero(peratom_adjoint_rho_sum, t_2, i_i_2);
  }
  #ifdef _OPENMP
  #pragma omp barrier
#endif
  fvec i_param_10 = (ONETYPE ? fvec::set1(this->param_cutoff) : fvec::mask_gather_double(t_0, ivec::setzero(), &this->param_cutoff));
  for (int t_3 = iifrom; t_3 < iito; t_3 += fvec::VL){
   ivec i_i_3 = ivec::set_consecutive() + ivec::set1(t_3);
   bvec t_4 = bvec::kand(bvec::full(), ivec::cmplt(i_i_3, ivec::set1(iito)));
   fvec i_px_4 = fvec::mask_gather<4>(fvec::undefined(), t_4, ivec::mul_fwidth(i_i_3), &x[0].x);
   fvec i_py_4 = fvec::mask_gather<4>(fvec::undefined(), t_4, ivec::mul_fwidth(i_i_3), &x[0].y);
   fvec i_pz_4 = fvec::mask_gather<4>(fvec::undefined(), t_4, ivec::mul_fwidth(i_i_3), &x[0].z);
   ivec i_ty_4 = ivec::mask_gather(ivec::undefined(), t_4, ivec::mul_fwidth(i_i_3), &x[0].w, 4);
   fvec i_a_5 = fvec::setzero();
   ivec i_i_6 = ivec::setzero();
   ivec t_5 = ivec::mask_gather(ivec::undefined(), t_4, i_i_3, numneigh, 4);
   for (;;){
    bvec t_6 = bvec::kand(t_4, ivec::cmplt(i_i_6, t_5));
    if (bvec::test_all_unset(t_6)) break;
    ivec i_a_7 = ivec::mask_gather(ivec::undefined(), t_6, ivec::mask_gather(ivec::undefined(), t_6, i_i_3, cnumneigh, 4) + (i_i_6),
firstneigh, 4);
    fvec i_px_8 = fvec::mask_gather<4>(fvec::undefined(), t_6, ivec::mul_fwidth(i_a_7), &x[0].x);
    fvec i_py_8 = fvec::mask_gather<4>(fvec::undefined(), t_6, ivec::mul_fwidth(i_a_7), &x[0].y);
    fvec i_pz_8 = fvec::mask_gather<4>(fvec::undefined(), t_6, ivec::mul_fwidth(i_a_7), &x[0].z);
    ivec i_ty_8 = ivec::mask_gather(ivec::undefined(), t_6, ivec::mul_fwidth(i_a_7), &x[0].w, 4);
    fvec i_dx_9 = ((i_px_4) - (i_px_8));
    fvec i_dy_9 = ((i_py_4) - (i_py_8));
    fvec i_dz_9 = ((i_pz_4) - (i_pz_8));
    fvec i_rsq_9 = ((((((i_dx_9) * (i_dx_9))) + (((i_dy_9) * (i_dy_9))))) + (((i_dz_9) * (i_dz_9))));
    bvec t_7 = bvec::kand(fvec::cmpnlt((i_rsq_9), (((i_param_10) * (i_param_10)))), t_6);
    if (bvec::test_any_set(t_7)){
```

Linnea

Kalman Filter

$$K_k := P_k^b H^T (H P_k^b H^T + R)^{-1};$$

$$x_k^a := x_k^b + K_k (z_k - H x_k^b);$$

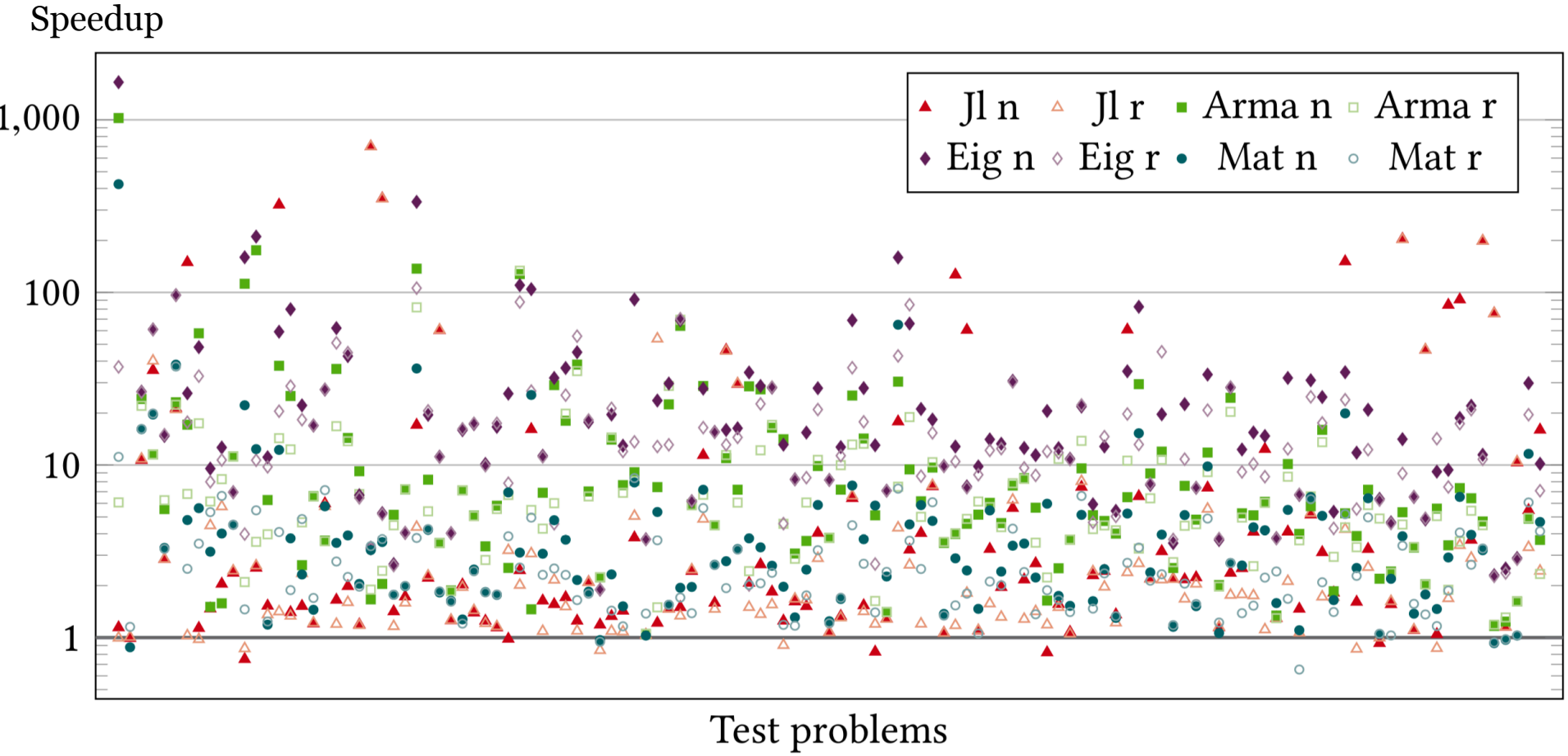
$$P_k^a := (I - K_K H)\, P_k^b$$

$K_k \in \mathbb{R}^{n \times m}$; $P_k^b \in \mathbb{R}^{n \times n}$, SPD;
$H \in \mathbb{R}^{m \times n}$, SPD; $R \in \mathbb{R}^{m \times m}$, SPSD;
$x_k^b \in \mathbb{R}^{n \times 1}$; $z_k \in \mathbb{R}^{m \times 1}$; $n = 400$; $m = 500$

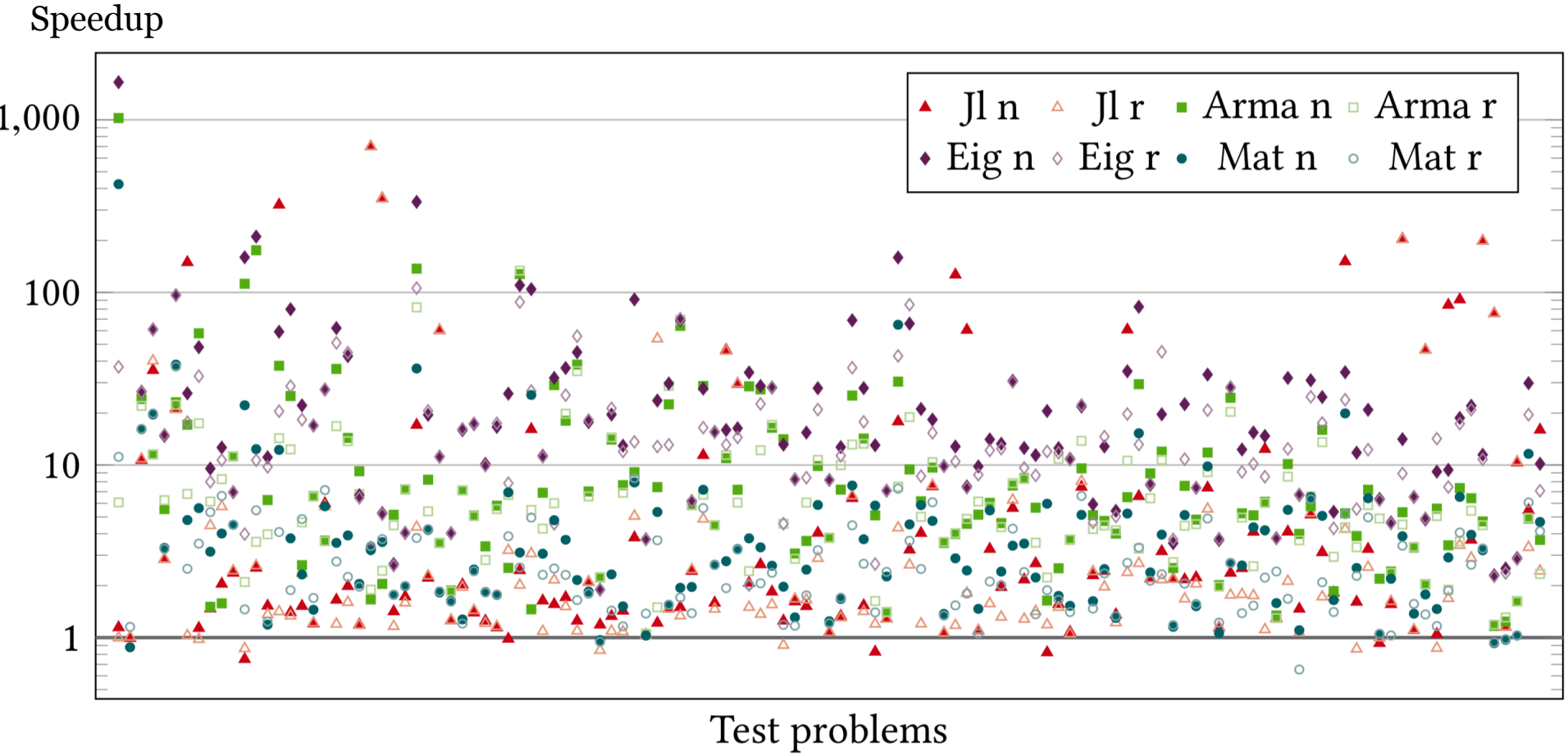
```
function algorithm0(ml0::Array{Float64,2}, ml1::Array{Float64,2}, ml2::Array{Float64,2}, ml3::Array{Float64,1}, ml4::Array{Float64,1})
    start::Float64 = 0.0
    finish::Float64 = 0.0
    Benchmarker.cachescrub()
    start = time_ns()

    # cost 5.34e+08
    # P_b: ml0, full, H: ml1, full, R: ml2, full, x_b: ml3, full, zk: ml4, full
    ml5 = Array{Float64}(undef, 400, 400)
    # tmp63 = (H P_b)
    symm!('R', 'L', 1.0, ml0, ml1, 0.0, ml5)

    # P_b: ml0, full, H: ml1, full, R: ml2, full, x_b: ml3, full, zk: ml4, full, tmp63: ml5, full
    ml6 = Array{Float64}(undef, 400, 400)
    # tmp62 = tmp63^T
    transpose!(ml6, ml5)

    # P_b: ml0, full, H: ml1, full, R: ml2, full, x_b: ml3, full, zk: ml4, full, tmp62: ml6, full
    # tmp19 = (R + (H tmp62))
    gemm!('N', 'N', 1.0, ml1, ml6, 1.0, ml2)

    # P_b: ml0, full, H: ml1, full, x_b: ml3, full, zk: ml4, full, tmp62: ml6, full, tmp19: ml2, full
    # (L64 L64^T) = tmp19
    LinearAlgebra.LAPACK.potrf!('L', ml2)

    # P_b: ml0, full, H: ml1, full, x_b: ml3, full, zk: ml4, full, tmp62: ml6, full, L64: ml2, lower_triangular
    ml7 = Array{Float64}(undef, 400, 400)
    blascopy!(400*400, ml6, 1, ml7, 1)
    # tmp70 = (tmp62 L64^-T)
    trsm!('R', 'L', 'T', 'N', 1.0, ml2, ml6)
```

Linnea

Linnea


Speedup
1,000
100
10
1
Jl n
Jl r
Arma n
Arma r
Eig n
Eig r
Mat n
Mat r
Test problems

Thank you


Speedup
1,000
100
10
1
Jl n
Jl r
Arma n
Arma r
Eig n
Eig r
Mat n
Mat r
Test problems