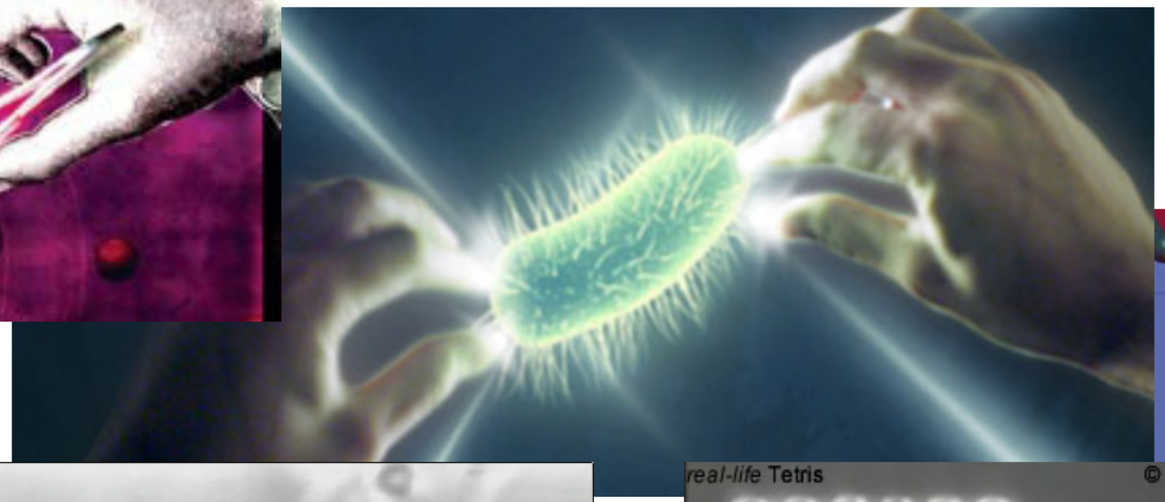
Určení sil působících na částici levitující v optické pinzetě

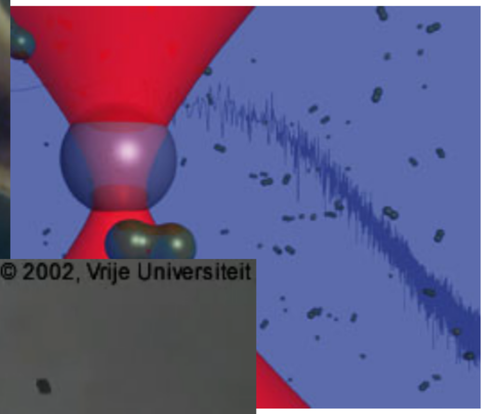
Martin Šiler

Optická pinzeta

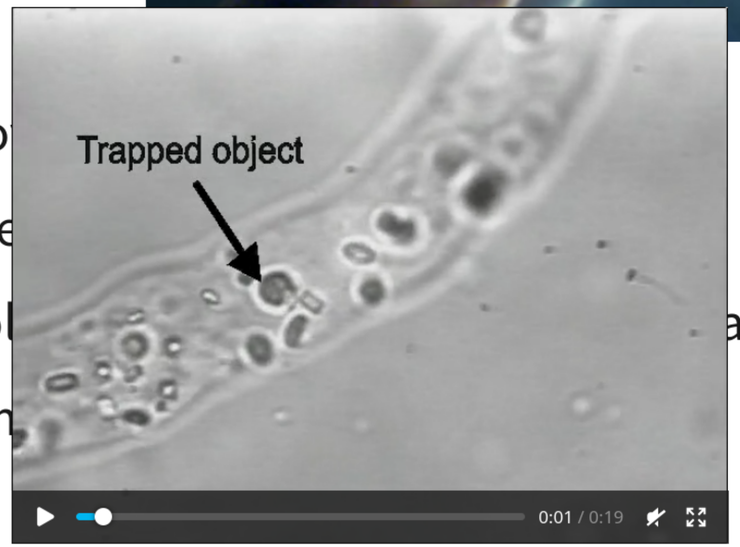
- silové účinky světla
- laser fokusovaný do mikroskopu
- biologie, rheologie, statistická fyzika, měření sil, tažný paprsek, …
- Simulink v realitě

Optická pinzeta



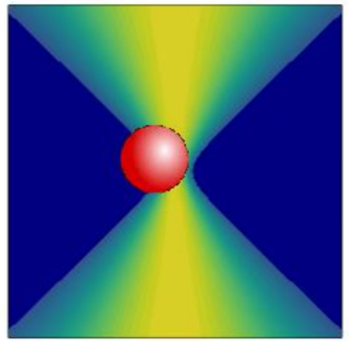


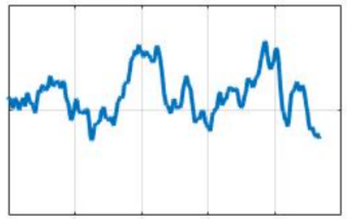
- silo
- lase
- biol a, m
- Sim



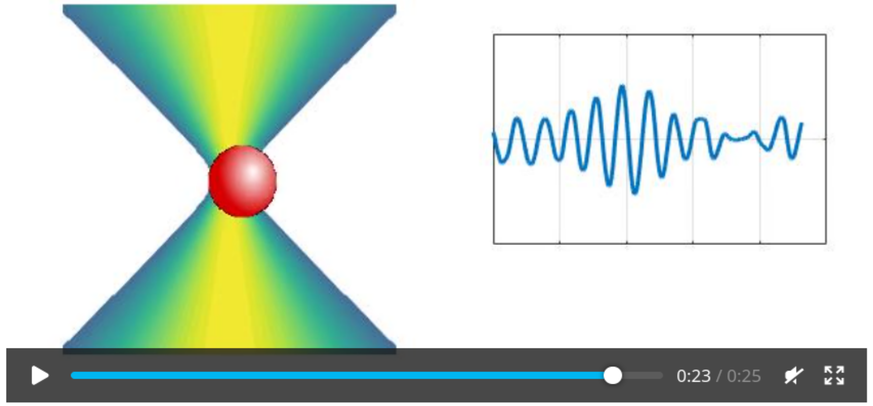
Optické levitace

Ve vodě

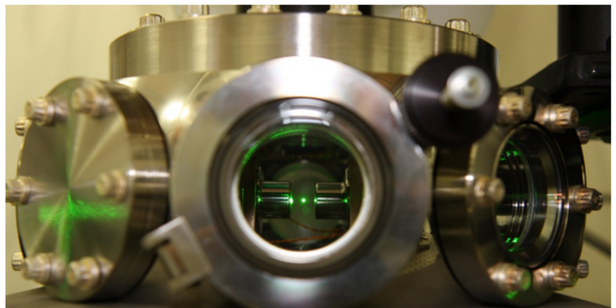




Ve vakuu

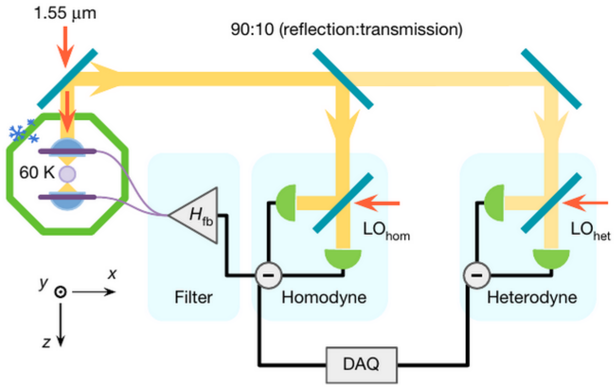


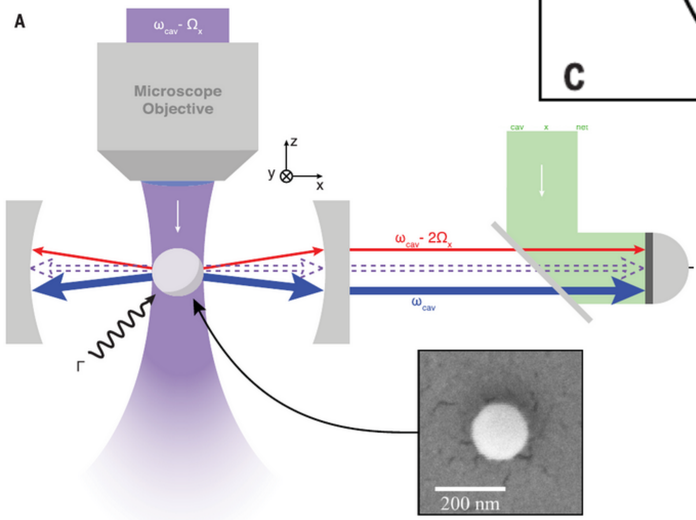
- harmonický oscilátor
- $m\ddot{x} + \gamma\dot{x} + m\Omega^2 x = \eta(t)$, random force $\langle \eta(t)\eta(t') \rangle = 2\gamma k_B T_{\text{eff}} \delta(t - t')$\$
- izolovaný od okolí
- Q factor až do 10^8

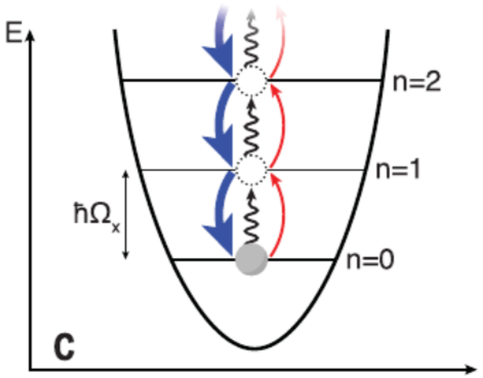


Kvantové makroskopické objekty

- $m\ddot{x} + \gamma\dot{x} + m\Omega^2 x = \eta(t)$, náhodná síla $\langle \eta(t)\eta(t') \rangle = 2\gamma k_B T_{\text{eff}} \delta(t - t')$\$
- tlumení a náhodné buzení jsou spojeny
- je možné zvýšit tlumení $\gamma\dot{x}$ a nechat stejné buzení

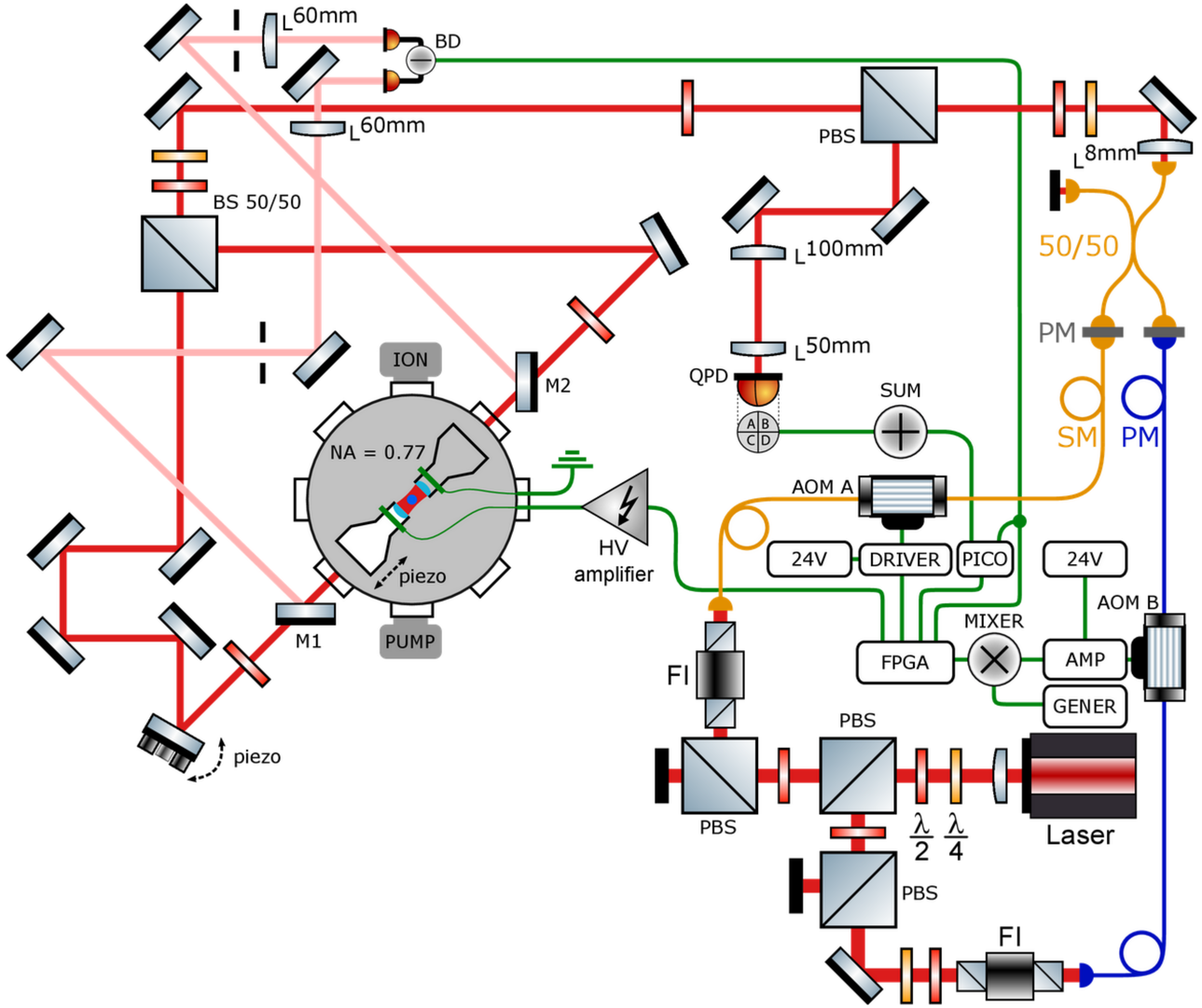






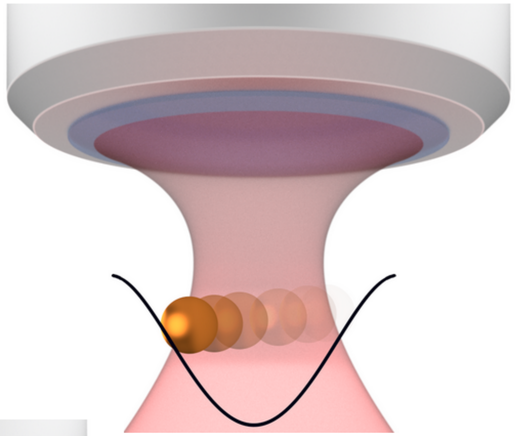
Delić U. et al., Science 367, 892–895 (2020)
Tebenjohans F. et al., Nature Vol 595, 378 (2021)
Kamba M. et al. Opt Express 30, 26716 (2022)

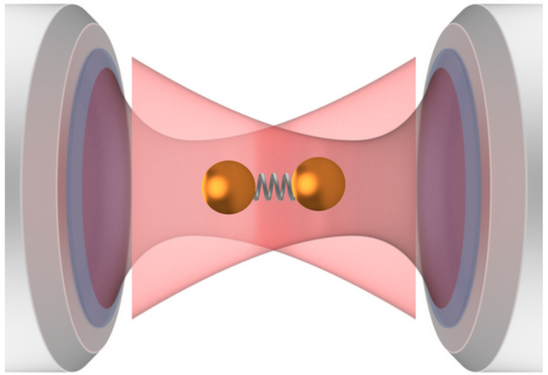
Experimentalní systém

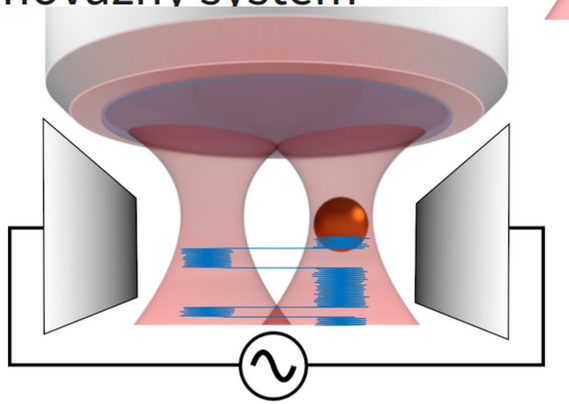


Characterizace levitující částice a pasti

- $m\ddot{x} + \gamma\dot{x} = F(x) + \eta(t; T, \gamma, m)$,
- **hmotnost** m : poloměr $\pm$ 10%, hustota $\pm$ 20%
- **tlumení** γ: model – tlak $\pm$ 30%, teplota, akomodace
- **optická síla** F: aberace, nepřesnosti, nelinearity
- **teplota** T: prostředí, částice se zahřeje, nerovnovážný systém



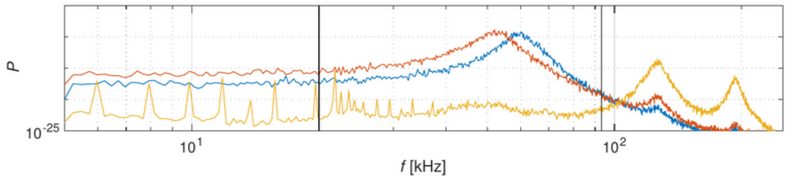




Spektrální hustota energie a Boltzmanovo rozdělení

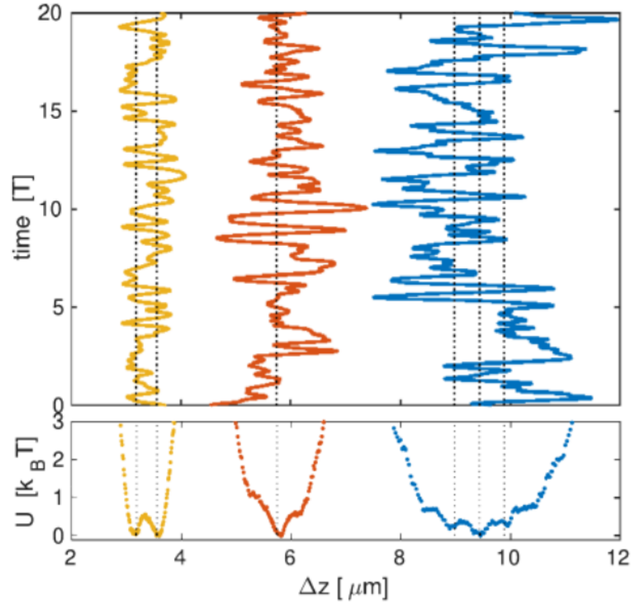
- Fourierova transformace trajektorie
- harmonický oscilátor

$$P_{xx}(\omega) = \frac{2k_{\mathrm{B}}T}{m} \frac{\Gamma}{\left(\omega^2 - \Omega_0^2\right)^2 + \Gamma^2\omega^2}$$



- rekonstrukce pravděpodobnosti
- potenciál

$$P(x) = \frac{1}{\mathcal{N}} \exp\left(-\frac{U(x)}{k_B T}\right)$$

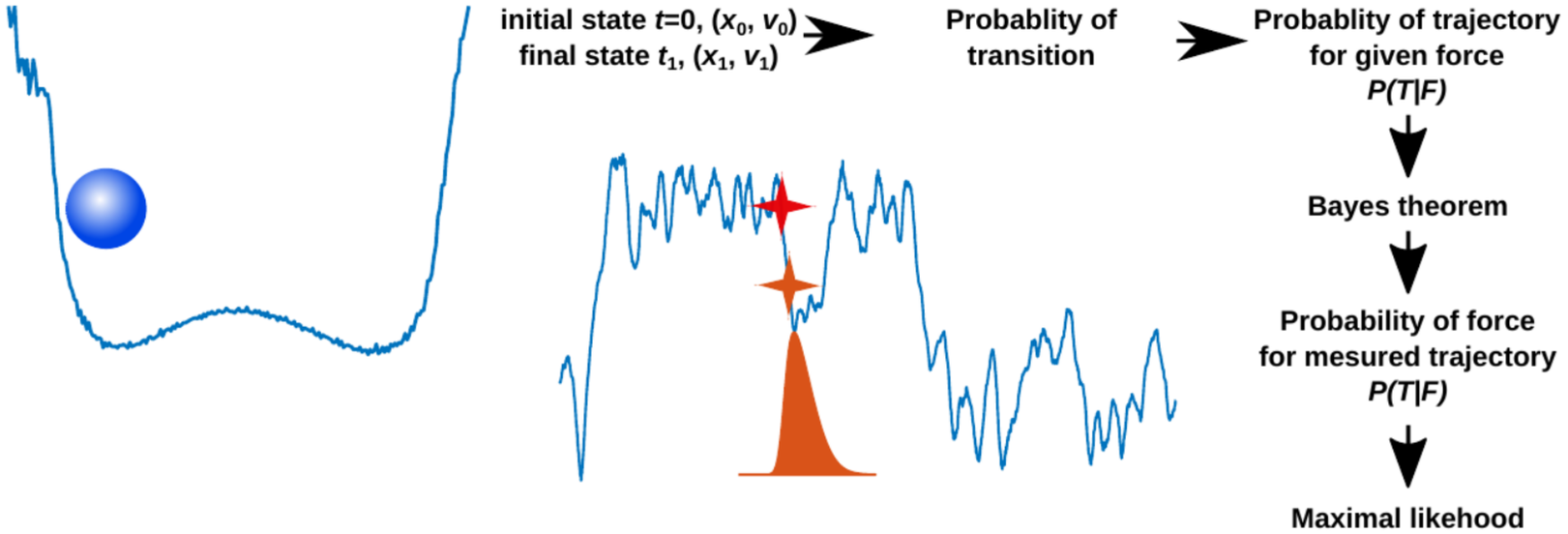


- Množství dat
- Stacionární stav
- PSD jen pro 1 typ síly
- Teplota / hmotnost

BEEPSIS

- Newtonovy zákony (pohybová rovnice)

$$m\ddot{x} + \gamma\dot{x} = F(x) + \eta(t; T, \gamma, m)$$



Bayes Estimation of Experimental Parameters in Stochatic Inertial Systems

BEEPSIS vstupy

- trajektorie $\mathcal{T}$
- měříme jen polohy
  - vyhnout se aproximacím rychlosti
  - setrvačnost + jen pozice $\Rightarrow$ ne-Markovovský proces
- chyby v detekci
- počet bodů trajektorie $(10^2 - 10^7)$
- samplovací frekvence

  ⋮
- **síla je konstantní během jednoho kroku**

[1] S. Tuerkcan et al., BIOPHYSICAL JOURNAL **102**, 2288 (2012).
[2] L. Perez Garcia et al., NATURE COMMUNICATIONS **9**, 5166 (2018).
[3] F. Ferretti et al., PHYSICAL REVIEW X **10**, 031018 (2020).

Matematika

- Probability of trajectory $\mathcal{T}$

$$P(\mathcal{T}|\boldsymbol{F},\Gamma,T) = \frac{1}{[(2\pi)^L \det\mathbf{C}]^{N/2}} \times \exp\left\{-\frac{1}{2}\sum_{n=x,y,\ldots}\sum_{i,j=1}^{L} \boldsymbol{m}_{i,n}(\mathbf{C}^{-1})_{ij}\boldsymbol{m}_{j,n}\right\},$$

- misfits

$$\boldsymbol{m_i} = \boldsymbol{x}_{i+1} - \boldsymbol{x}_i\left(1+\mathrm{e}^{-\Gamma\tau}\right) + \boldsymbol{x}_{i-1}\mathrm{e}^{-\Gamma\tau} - \frac{\boldsymbol{F(x_i)}\tau}{m\Gamma}\left(1-\mathrm{e}^{-\Gamma\tau}\right)$$

- covariance matrix

$$\mathbf{C} = \begin{pmatrix} a & b & 0 & & \cdots & 0 \\ b & a & b & & & \\ 0 & b & a & b & & \\ & & \ddots & \ddots & \ddots & \\ & & & & & \\ \vdots & & & & & \\ \vdots & & & b & a & b \\ 0 & & & & b & a \end{pmatrix}$$

$$\begin{aligned} a &= 2\frac{k_B T}{m\Gamma^2}\left[\Gamma\tau - 1 + (\Gamma\tau+1)\,\mathrm{e}^{-2\Gamma\tau}\right], \\ b &= \frac{k_B T}{m\Gamma^2}\left[1 - 2\Gamma\tau\mathrm{e}^{-\Gamma\tau} - \mathrm{e}^{-2\Gamma\tau}\right]. \end{aligned}$$

- Detection error $\mathbf{C} \to \mathbf{C} + \boldsymbol{\Psi}$
- Maximum likehood estimation $\to$ minimization

$$\mathcal{S} = -\log P\left(\mathcal{T}|\boldsymbol{F},\Gamma,T\right) = \frac{LN}{2}\log 2\pi + \frac{N}{2}\log\det\boldsymbol{C} + \frac{1}{2}\sum_{n=x,y,\ldots}\sum_{i,j=1}^{L}\boldsymbol{m}_{i,n}(\mathbf{C}^{-1})_{ij}\boldsymbol{m}_{j,n}$$

[1] D. Sivia et al., (Oxford University Press, 2006).

MATLAB

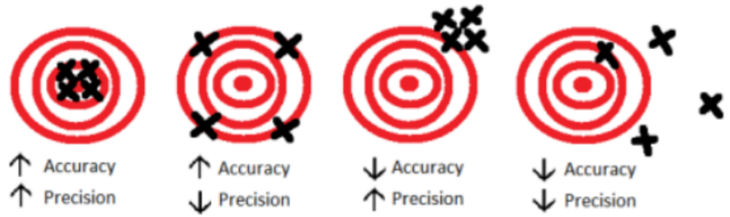
- problém: Kovarianční matice $(N-2) \times (N-2)$
  - $100 < N < 10^7$
  - 3 (5) diagonál **pásová matice**
  - potřebujeme $\log\det\mathbf{C}$ a $\mathbf{m.}'(\mathbf{C}\backslash\mathbf{m})$
- řídké matice (`sparse`)
  - inicializace matice $\gg$ výpočet
- CUDA + `sparse`
  - opět problémy i inicializací a transferem dat
- Choleského dekompozice $\mathbf{C} = \mathbf{L}'\mathbf{L}$ (trojúhelníkové matice)
  - choleski + pásová matice = pásové matice
  - MATLAB nemá funkce
  - MATLAB central existuje vhodná (pomalé)
  - přepsat do C

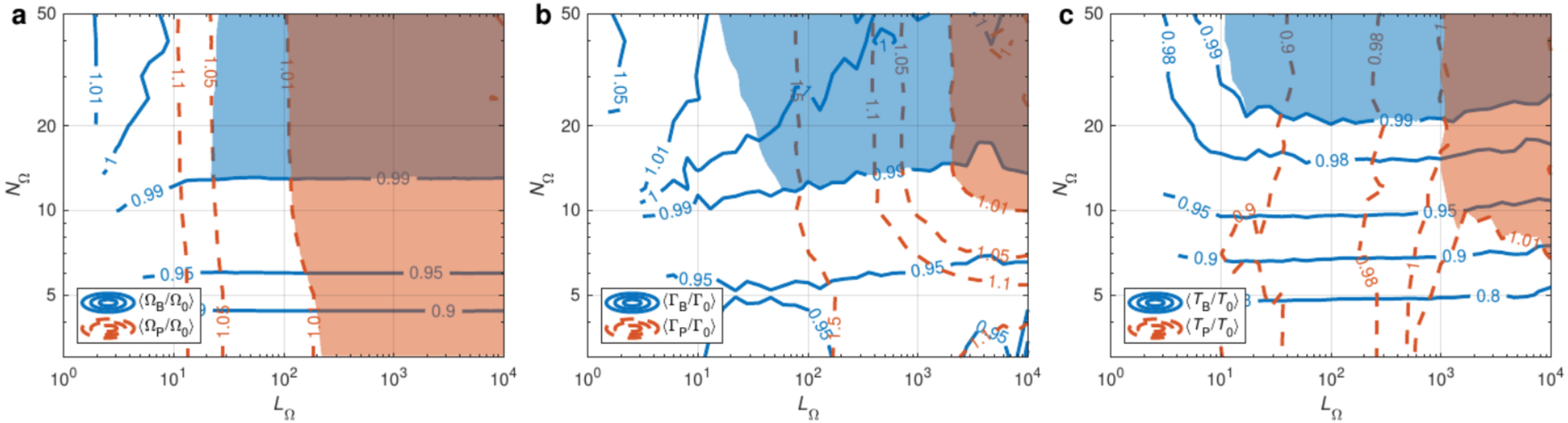
$$\mathbf{C} = \begin{pmatrix} a & b & 0 & & \cdots & 0 \\ b & a & b & & & \\ 0 & b & a & b & & \\ & \ddots & \ddots & \ddots & \ddots & \\ & & & & & \\ \vdots & & & & & \\ & & & b & a & b \\ 0 & & & & b & a \end{pmatrix}$$

```
void mexFunction(int nlhs, mxArray *plhs[], int nrhs, const mxArray *prhs[])
{
/* Argument checks */
    if( nlhs > 4 ) {
        mexErrMsgTxt("Too many outputs");
    }
    if( nrhs != 2 ) {
        mexErrMsgTxt("Need exactly 2 inputs");
    }

    //cov matrix
    if( !(mxIsDouble(prhs[0]) ) || mxIsSparse(prhs[0]) || mxIsComplex(prhs[0]) || (mxGetNumberOfDimensions(prhs[0]) > 2))
    {
        mexErrMsgTxt("1st argument must be real double full matrix");
    }
    //rhs vector/matrix
    if( !(mxIsDouble(prhs[1]) ) || mxIsSparse(prhs[1]) || mxIsComplex(prhs[1]) || (mxGetNumberOfDimensions(prhs[1]) > 2) )
    {
        mexErrMsgTxt("2nd argument must be real double full vector/matrix");
    }

    const mwSize *sz = mxGetDimensions(prhs[0]);
    const int B = sz[0];
    const int N = sz[1];

    if (N<1)
        mexErrMsgTxt("Symmetric band array size must be at least 1");
    if (B<1)
        mexErrMsgTxt("Symmetric band array number of bands must be at least 1");
    if (B>N)
        mexErrMsgTxt("Symmetric band array number of bands must <= array size");


    const mwSize *sz2 = mxGetDimensions(prhs[1]);
    if (sz2[0] != N)
    {
        mexErrMsgTxt("Symmetric band array size must be the same as number of rows of rhs");
    }

    const int RHC = sz2[1];


    mwSize ss[2];
    ss[0] = 1; ss[1] = 1;
    mxArray * mLogDet = mxCreateNumericArray(2, ss, mxDOUBLE_CLASS, mxREAL);

    ss[1] = RHC;
    mxArray * mSol    = mxCreateNumericArray(2, ss, mxDOUBLE_CLASS, mxREAL);

    ss[0] = B; ss[1] = N;
    mxArray *mChol    = mxCreateNumericArray(2, ss, mxDOUBLE_CLASS, mxREAL);

    ss[0] = N; ss[1] = 1;
    mxArray *mVec     = (nlhs<4) ? NULL: mxCreateNumericArray(2, ss, mxDOUBLE_CLASS, mxREAL);


    const double *bandarray = mxGetDoubles(prhs[0]);
    const double *rharray  = mxGetDoubles(prhs[1]);
    double *sol    = mxGetDoubles(mSol);
    double *logdet = mxGetDoubles(mLogDet);
    double *chol   = mxGetDoubles(mChol);
    memcpy(chol, bandarray, N*B*sizeof(double));

    //calculate Choleski decomposition of covariance matrix
    symbandchol(N, B, chol);
    //calculate log det of covariance matrix
    logdet[0] = logdetC(N, B, chol);

    //calculate 1st sum of mC^{-1}m'
    sol[0]    = solve_sb_rhs(N, B, chol, &rharray[0], mVec ? mxGetDoubles(mVec) : NULL);
```

BEEPSIS přesnost

- Simulated trajectories
- $F = -m\Omega_0^2 x$
- Ω_0, Γ_0, T_0 randomly selected
- 10^4 trajectories simulated
- number of points per period + trajectory length

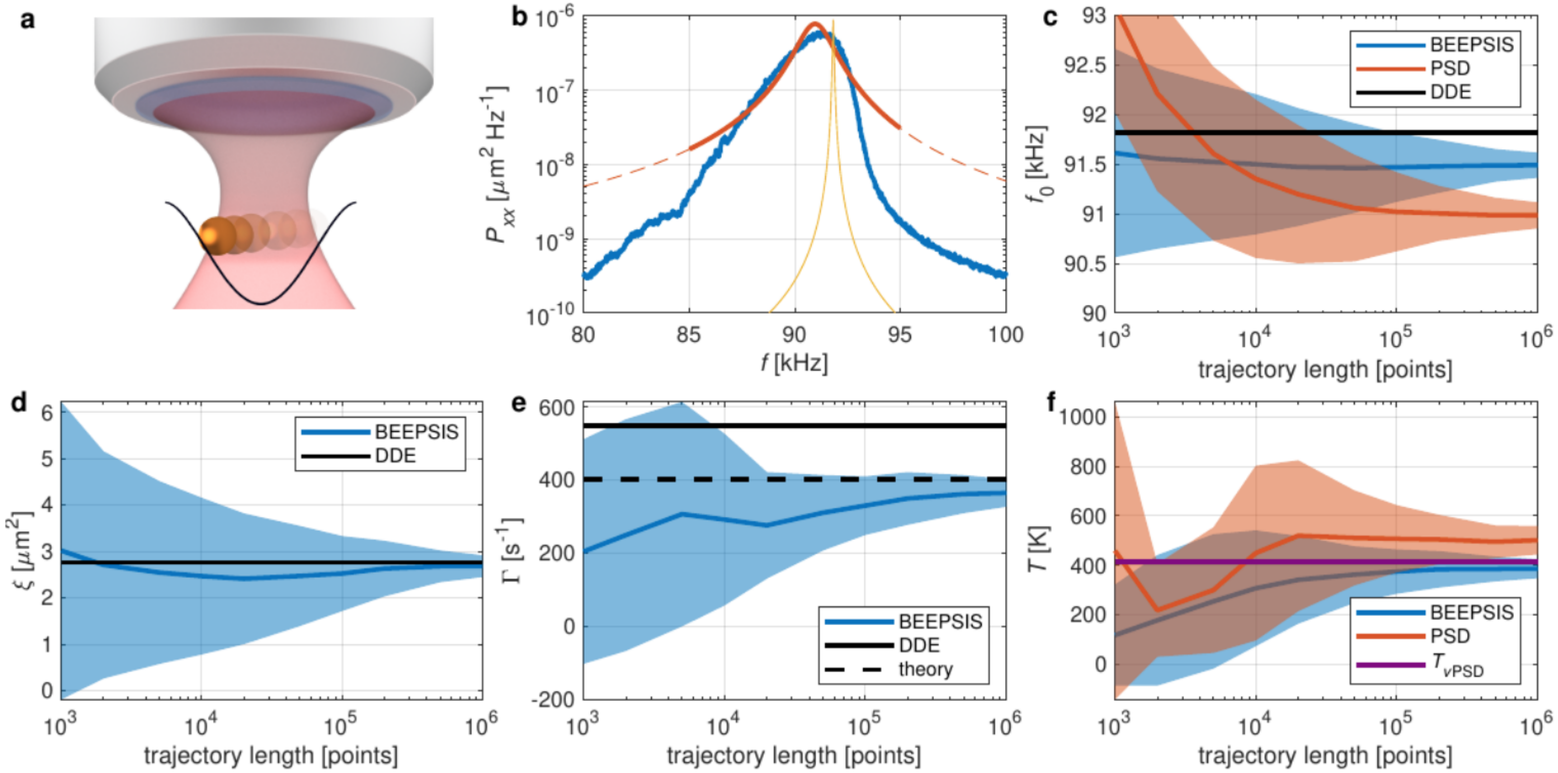




[1] M. Šiler et al., Physical Review Applied **19**, 064059 (2023).

BEEPSIS optická pinzeta

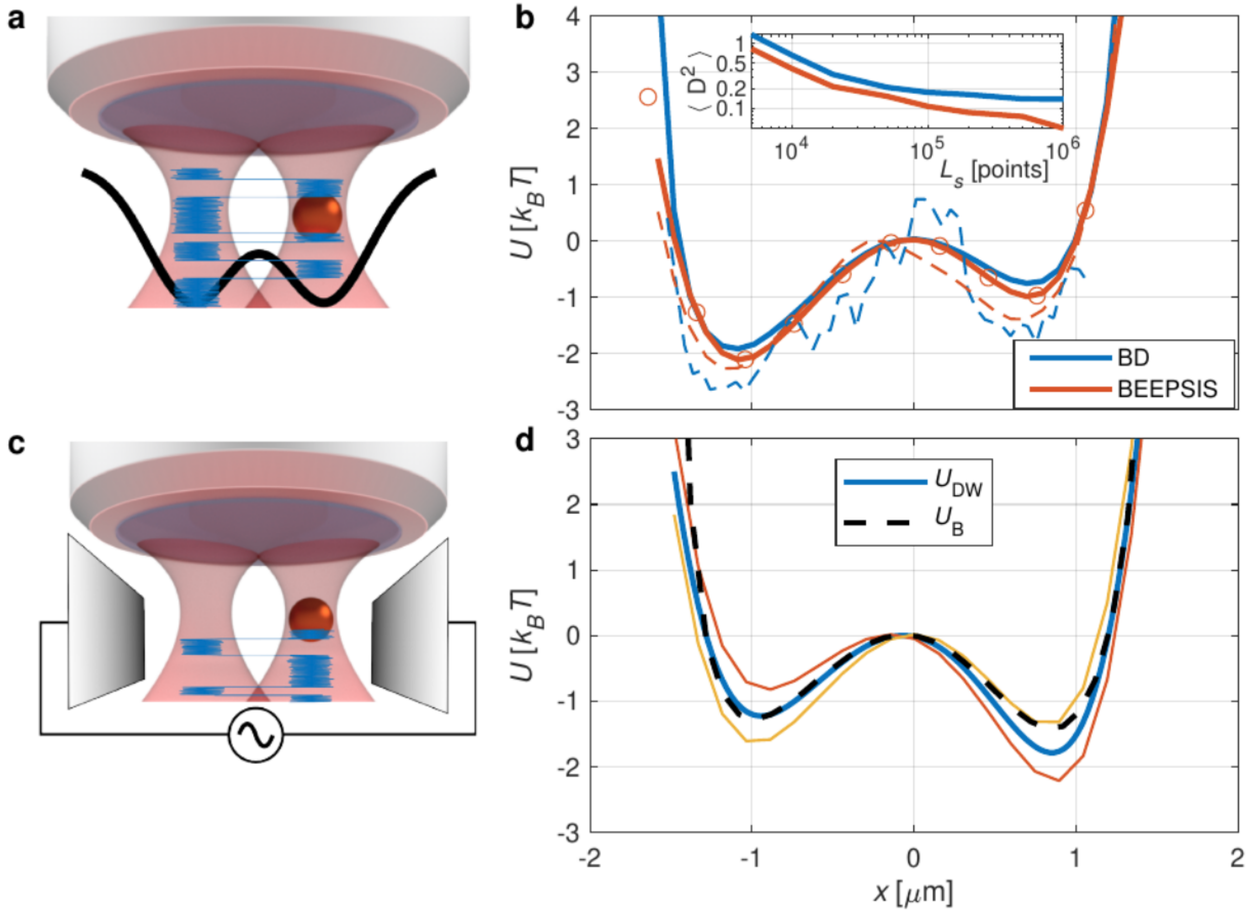
- Duffing force $F = -m\Omega_0^2\left(1 - \xi x^2\right)$



- d = 170 nm
- NA = 0.7
- λ = 1064 nm
- p = 0.1 mBar
- f_{sample}= 1.73 MHz
- 60 trajectories
- 10^6 points each

[1] M. Šiler et al., Physical Review Applied **19**, 064059 (2023).

BEEPSIS dvojitá jáma



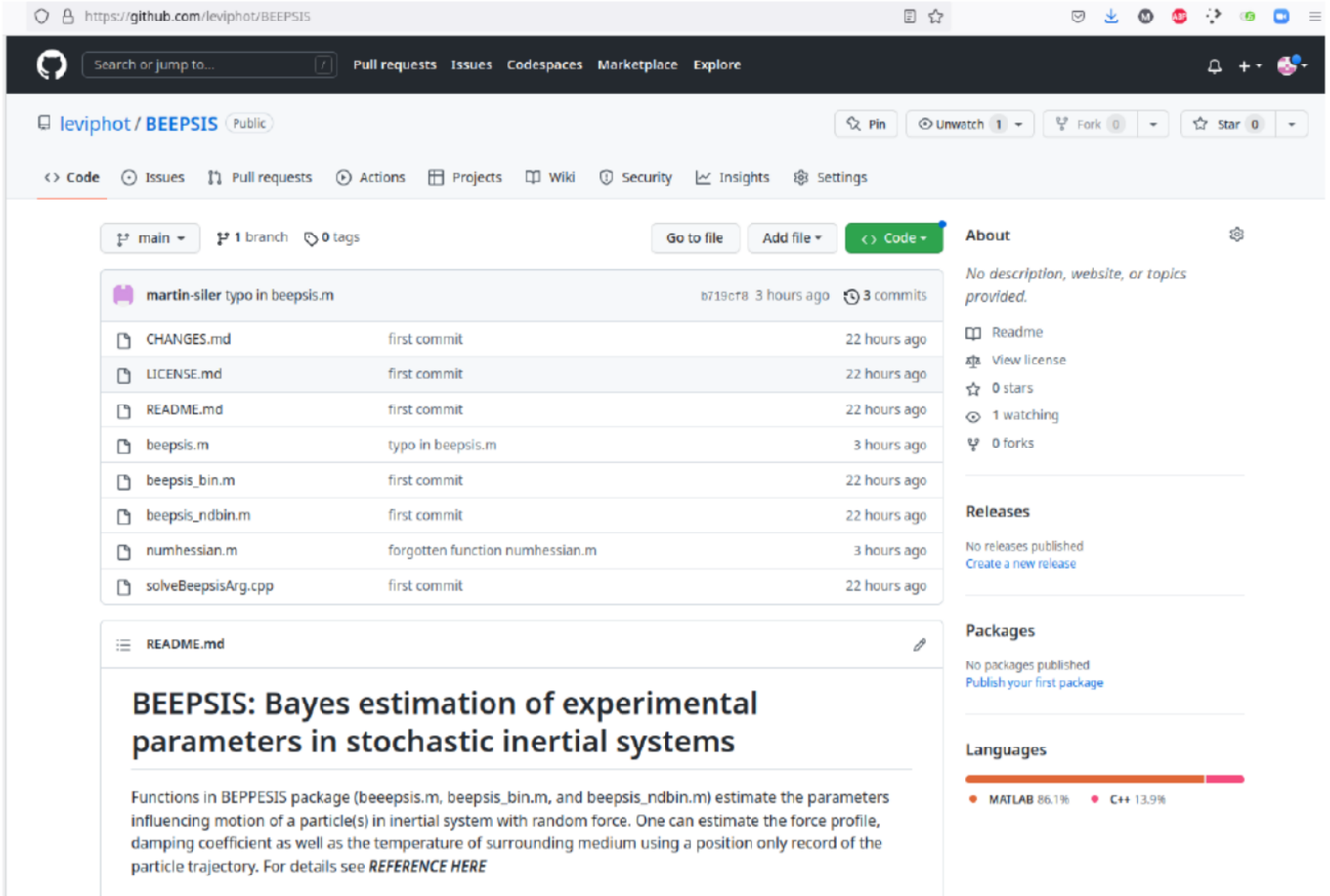
- dual beam trap
- d = 600 nm
- force in 11 points
  + spline interpolation
- T_{eff} = 229 K $\rightarrow$ bigger particle
- $\Gamma_i > \Gamma_{\mathrm{theory}}$
  $\rightarrow$ different pressure (25%)

- time dependent force $f = 250$ Hz
- amplitude of driving new parameter

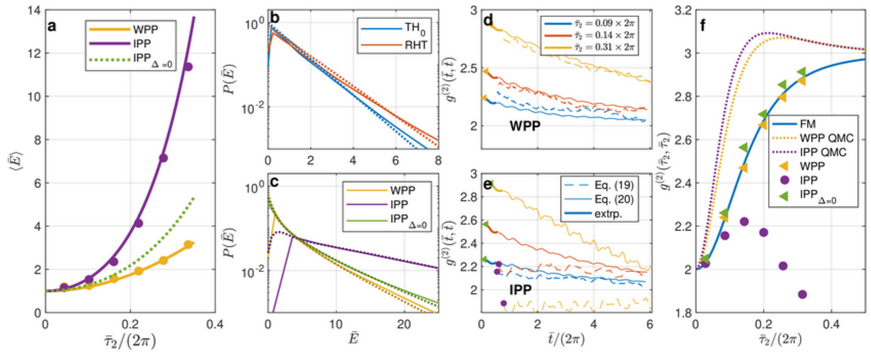
[1] M. Šiler et al., Physical Review Applied **19**, 064059 (2023).

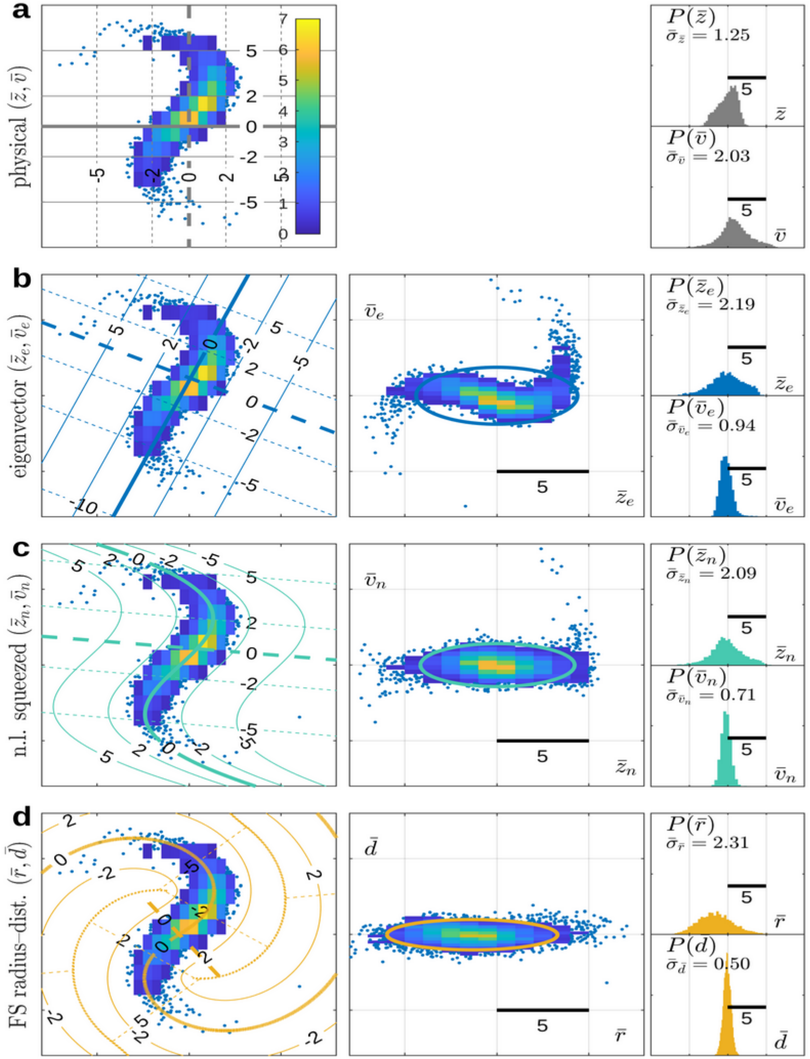
BEEPSIS kód

- https://github.com/leviphot/BEEPSIS



MATLAB zajímavosti

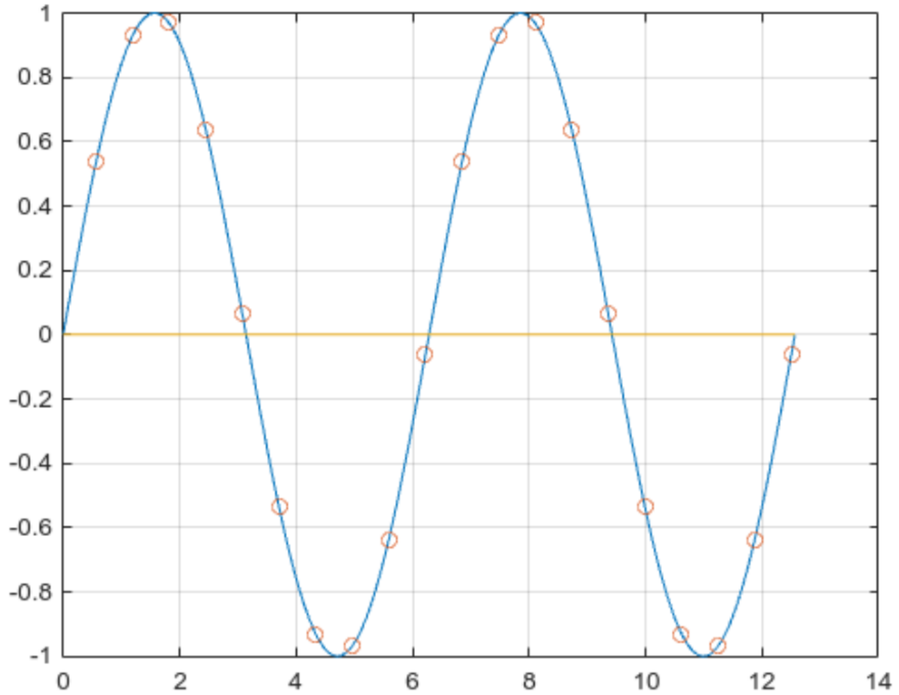




MATLAB zajímavosti

Barvy

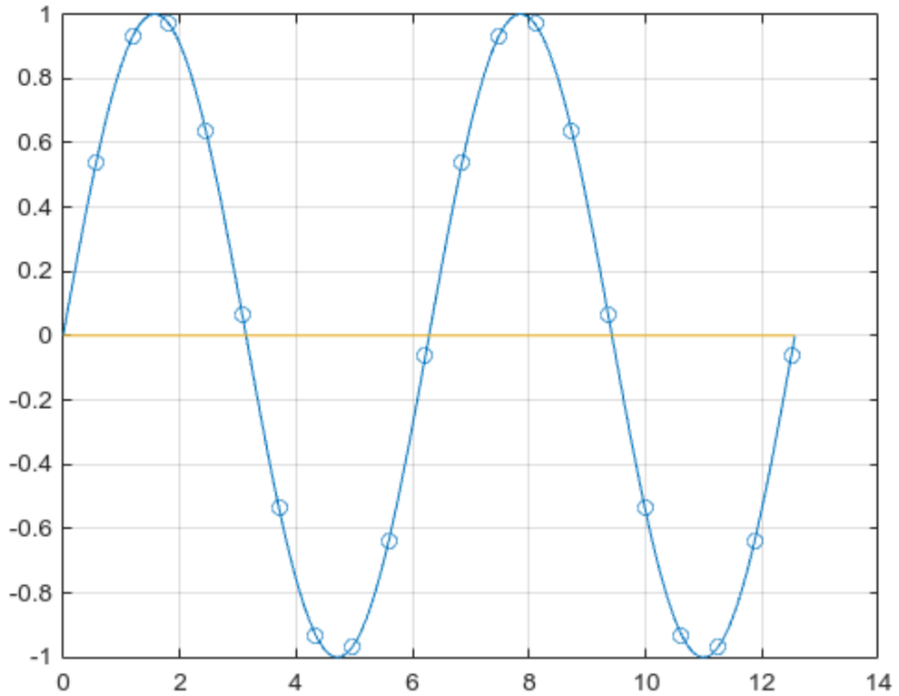
```
x = linspace(0, 4*pi, 201);
y = sin(x);

plot(x, y)
hold on
plot(x(10:10:end),y(10:10:end), 'o' )
plot(x, 0*x)
hold off
```



MATLAB zajímavosti

Barvy

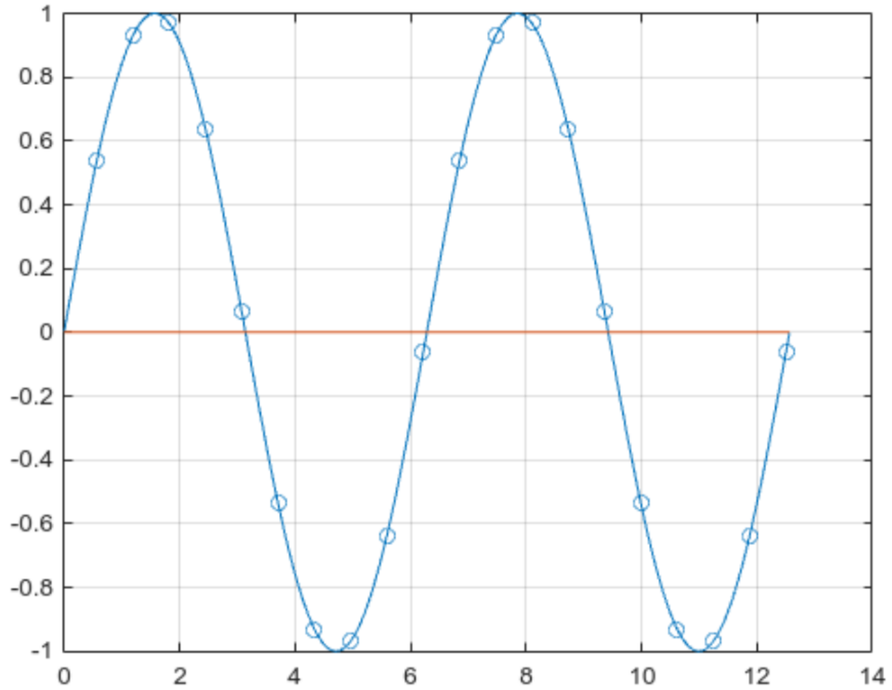
```
clist = get(gca, 'ColorOrder');

plot(x, y)
hold on
plot(x(10:10:end),y(10:10:end), 'o' , 'Color',clist(1,:))
plot(x, 0*x)
hold off
```



MATLAB zajímavosti

Barvy

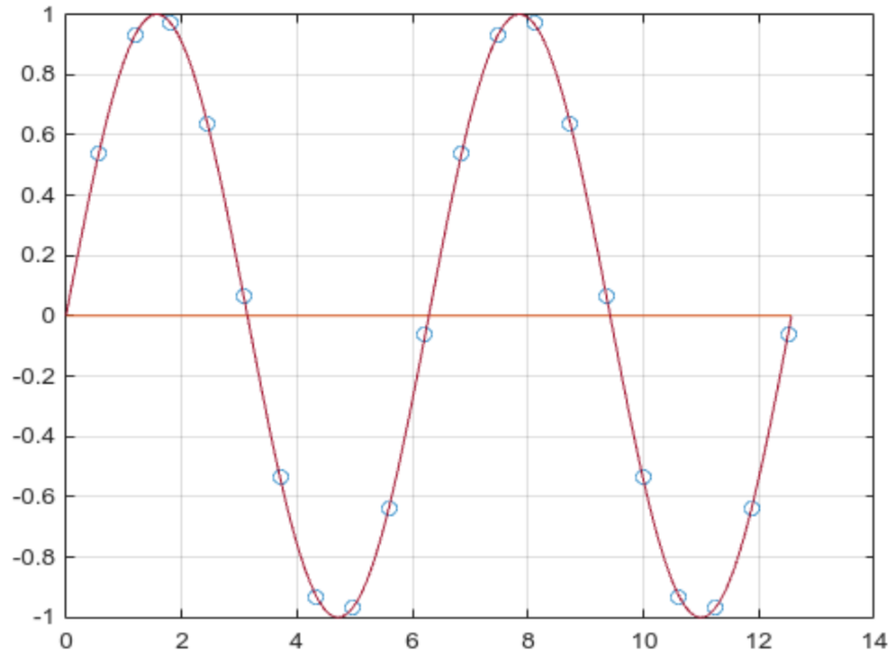
```
ax = gca;
plot(x, y)
hold on
ax.ColorOrderIndex = ax.ColorOrderIndex-1;
plot(x(10:10:end),y(10:10:end), 'o' )
plot(x, 0*x)
hold off
```



MATLAB zajímavosti

Barvy

```
ax = gca;
hold on
ax.ColorOrderIndex = 7;
plot(x, y)
```




```
ax.ColorOrderIndex = ax.ColorOrderIndex-1;
```

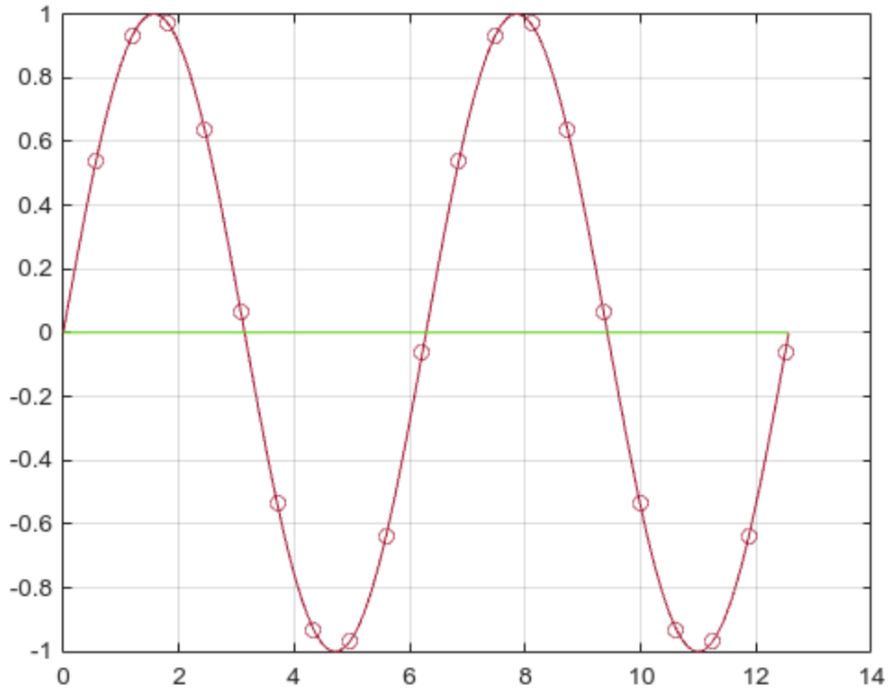
```
Error setting property 'ColorOrderIndex' of class 'Axes':
Value should be an integer greater than 0
```

```
plot(x(10:10:end),y(10:10:end), 'o' )
plot(x, 0*x)
```

MATLAB zajímavosti

Barvy

```
ax = gca;
hold on
axTool.setColor(7)
plot(x, y)
axTool.prevColor;
plot(x(10:10:end),y(10:10:end), 'o' )
axTool.extendCList
plot(x, 0*x, 'Color',axTool.Color(11))
hold off
```



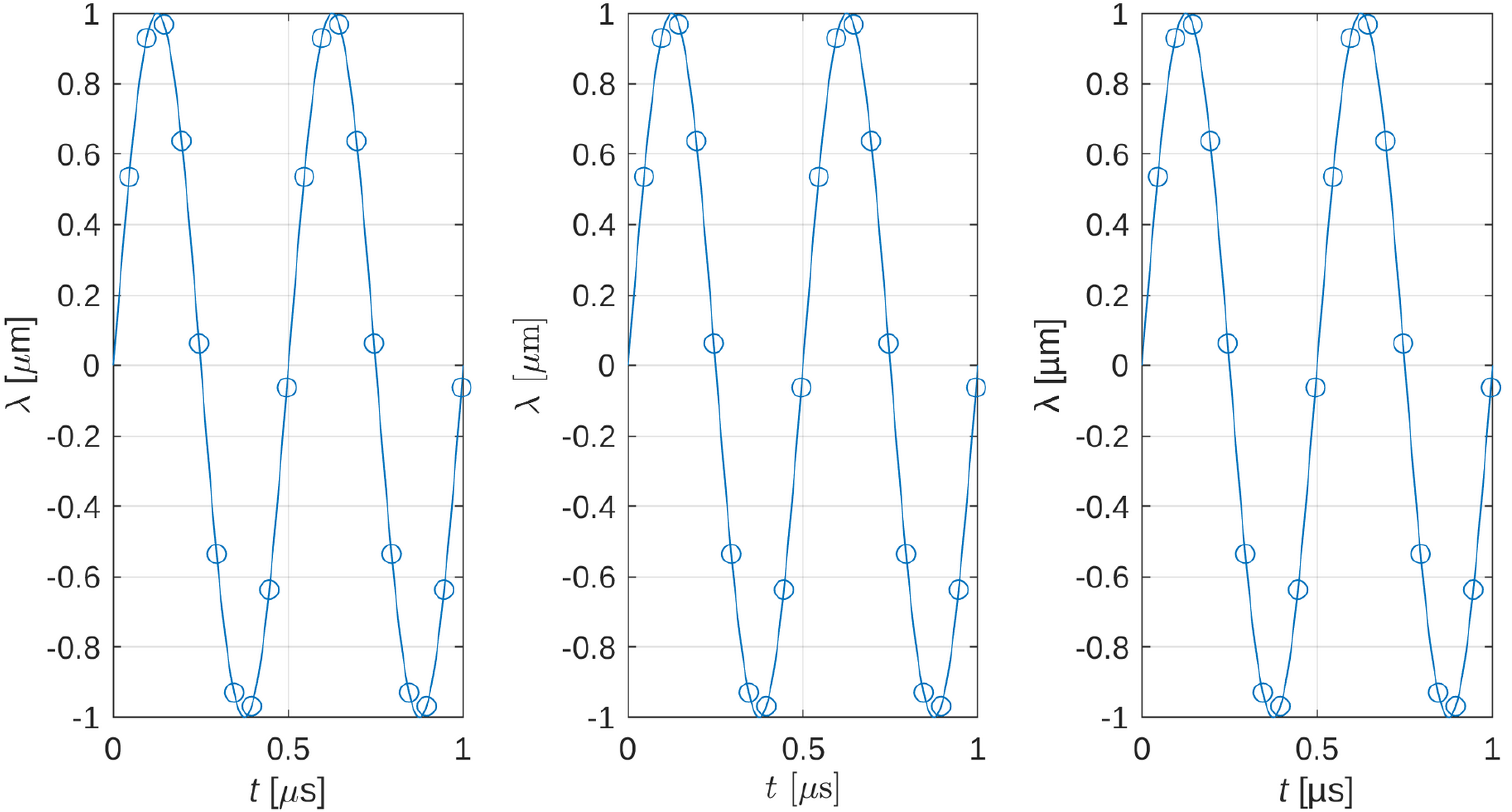
MATLAB zajímavosti

Barvy

```
classdef axTool
    methods(Static = true)
        function clist = extendCList(ax, repeats)
            if nargin == 0
                ax = gca;
            end
            ax.ColorOrder = [ax.ColorOrder;
                0.8,0.8,0.05;
                0.07, 0.62, 1;
                1, 0.41,0.16;
                0.39, 0.83, 0.07;
                0.72, 0.27,1;
                0.06,1,1;
                1, 0.07, 0.65];
            if nargin == 2
                ax.ColorOrder = repmat(ax.ColorOrder, repeats,1);
            end
            ax.NextPlot = 'add';
            ax.Box      = 'on';
            if nargout >= 1
                clist = ax.ColorOrder;
            end

        end

        function prevColor(ax)
            if nargin == 0
                ax = gca;
            end
            if ax.ColorOrderIndex == 1
                ax.ColorOrderIndex = size(ax.ColorOrder,1);
            else
                ax.ColorOrderIndex = ax.ColorOrderIndex -1;
            end
        end
        function nextColor(ax) ...
        function setColor(ax, index)
            if nargin == 1
                index = ax;
                ax = gca;
            end
            index = mod(index - 1,  size(ax.ColorOrder,1)) + 1;
            ax.ColorOrderIndex = index;
        end
        function resetColor(ax) ...

        function clr = Color(index, ax) ...

        function m = Marker(index)
            persistent last;

            markers = 'osd^v><+*ph._|';

            if nargin == 0 || isempty(index)
                m = markers;
                return
            end
            if isempty(last)
                last = 0;
            end
            if index < 1
                index = last + 1;
            end
            ii = mod(index-1, numel(markers)) + 1;
            m = markers(ii);
            last = ii;
        end
    end
end
```

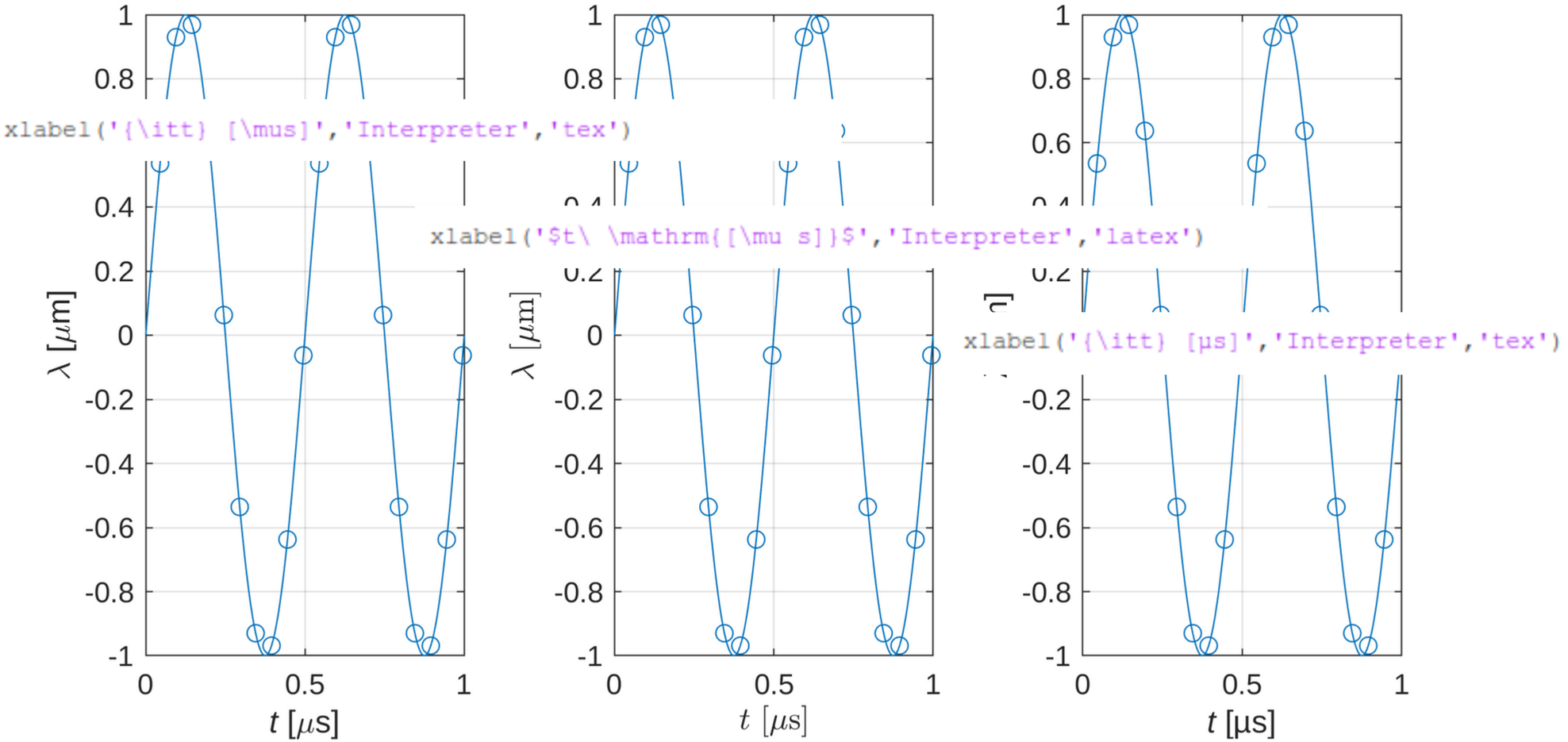
MATLAB zajímavosti

Písmo



MATLAB zajímavosti

Písmo



Děkuji za pozornost

Team members

- Oto Brzobohatý
- Pavel Zemánek
- Petr Jákl
- Alexandr Jonáš
- Tereza Zemánková
- Vojtěch Liška
- Martin Duchaň
- Tadeáš Maňka
- Jan Klusáček
- Erik Búš
- Martin Šarbort
- Lenka Čermáková

This work was supported by GACR (GA23-06224S), AVCR (Praemium Academiae), and MSMT (CZ.02.01.01/00/22_008/0004649)