MELODY/MATLAB

OBJECT-ORIENTED MODEL OF GRAVITATIONAL-WAVE INTERFEROMETERS USING MATLAB

Raymond G. Beausoleil

Stanford University/Hewlett-Packard Laboratories
13837 175th Pl. NE, Redmond, WA 98052–2180

beausol@hpl.hp.com

August 2000

LIGO-G000224-00-D

MELODY/MATLAB OVERVIEW

- Goals and features
- Propagation model
- Object-level features
  - Interferometer configurations
  - Mirror physics: thermal loading, position, orientation
  - Four-stage resonator length pseudolocking
- Script-level features
  - Modulation schemes
  - Mirror parameters: thermal, position, orientation
  - Full interactive MATLAB functionality
- Milestones

ACKNOWLEDGMENTS

- *STANFORD*
  - Eric Gustafson
  - Marty Fejer
- *Glasgow*
  - Ken Strain
- *UCF*
  - Guido Mueller
- *MIT*
  - Ryan Lawrence
  - Daniel Sigg

MELODY/MATLAB GOALS

- Provide an easily usable, flexible multiplatform framework for LIGO I/II calculations and simulations
- Allow users to write scripts to drive simulations tailored to their needs (post-processing, graphics, numerical analysis)
- Easily include physical effects in mirrors: thermal lensing, thermoelastic surface deformation, curvature mismatch, diffraction
- Allow translation to a lower-level language for performance
- Provide a simple interface to industry-standard software for modeling control systems (SIMULINK)

MELODY/MATLAB FEATURES

- MATLAB classes for fields, mirrors, interferometers, and detectors; driven by user-written scripts
- Prebuilt LIGO I/II (& GEO 600) configurations
  - Power, signal, and dual recycling
  - Arbitrary modulation schemes
  - Resonator length pseudolocking for self-contained simulations
- Mirror physics
  - Mirror surface/laser wavefront curvature mismatch
  - Thermal lensing due to bulk and coating absorption
  - Thermoelastic distortions of the reflecting surface
  - Aperture diffraction

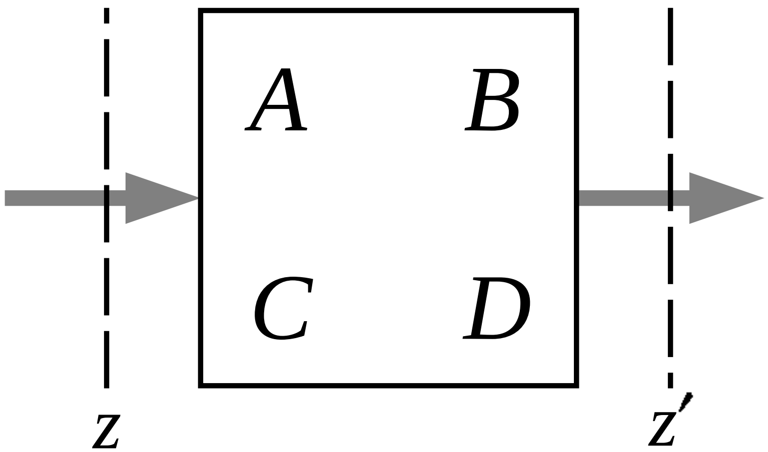
NEW MELODY/MATLAB FEATURES

- Arbitrary number of transverse modes allowed
- Aperture diffraction implemented (but should be tested against the FFT model)
- Four-stage pseudolocker implemented: FPI $\longrightarrow$ BS $\longrightarrow$ PRM $\longrightarrow$ SRM
- Hello-Vinet thermoelastic surface deformation added to mirror class (Saint Venant correction ignored)
- Precomputation of all matrix operators allows rapid initialization using MATLAB's LOAD command

MELODY/MATLAB LIMITATIONS

- Models thermal loading due to TEM_{00} absorption only, summed over all frequency components
- Beamsplitter thermoelastic surface deformations not yet implemented (Algor)
- Non-normal incidence angle not generally implemented (Algor)
- Transient thermal loading not yet implemented (calculations complete)

FORWARD PROPAGATION: HUYGENS-FRESNEL INTEGRAL



$$\mathbf{E}(\mathbf{r}, t) \equiv \mathrm{Re}\left\{\boldsymbol{\epsilon} E(\mathbf{r}) e^{i(kz-\omega t)}\right\}$$

$$\nabla_{\perp}^2 E(\mathbf{r}) + i2k\frac{\partial}{\partial z}E(\mathbf{r}) = 0$$

$$E(x, y, z) = \int_{\mathcal{A}_1} dx'\, dy'\, K(x, y; x', y') E(x', y', z') \equiv \hat{K}[E(x', y', z')]$$

$$K(x, y; x', y') =$$

$$\frac{1}{i\lambda B} \exp\left\{ i\frac{\pi}{\lambda B}\left[A(x'^2 + y'^2) - 2(x'x + y'y) + D(x^2 + y^2)\right]\right\}$$

Forward and backward unperturbed eigenmodes:

$$\gamma_{mn} u_{mn}(x, y, 0) = \int_{\mathcal{A}_1} dx'\, dy'\, K_0(x, y; x', y') u_{mn}(x', y', 0)$$

$$\gamma_{mn}^{\dagger} u_{mn}^{\dagger}(x, y, 0) = \int_{\mathcal{A}_1} dx'\, dy'\, K_0^{\dagger}(x, y; x', y') u_{mn}^{\dagger}(x', y', 0)$$

Biorthogonality relation (Siegman), satisfied discretely:

$$\int_{\mathcal{A}_1} dx\, dy\, u_{mn}^{\dagger}(x, y, z) u_{m'n'}(x, y, z) = \delta_{mm'} \delta_{nn'}$$

Expand intracavity field:

$$E(x, y, z, t) = \sum_{mn} E_{mn}(t) u_{mn}(x, y, z)$$

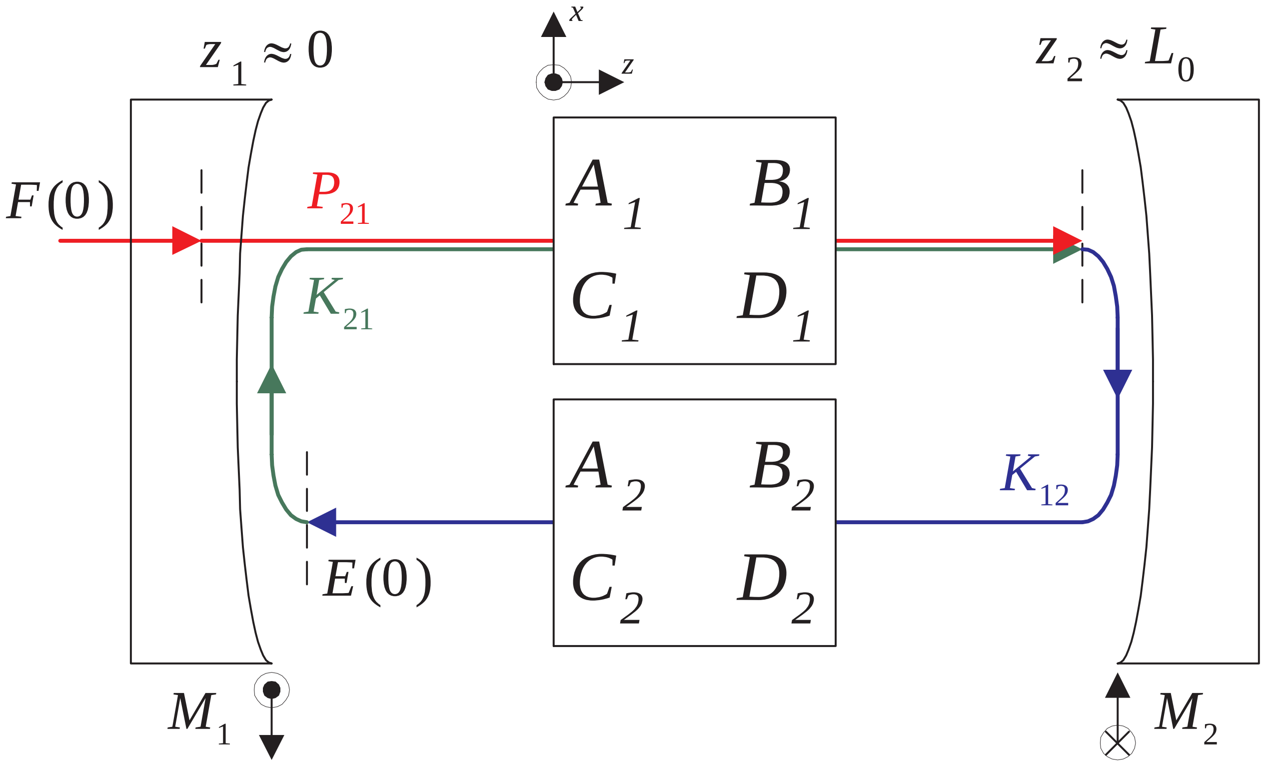
PROPAGATOR MATRIX ELEMENTS

Calculate $K_{mn;m'n'}(t)$ as the matrix element of the fully perturbed forward propagator (from reference plane $\mathcal{A}_1$ to reference plane $\mathcal{A}_2$) in the basis of the unperturbed eigenmodes:

$$K_{mn;m'n'}(t) = \int_{\mathcal{A}_2} dx\,dy \int_{\mathcal{A}_1} dx'\,dy' \times u^{\dagger}_{mn}(x,y) K(x,y;x',y';t) u_{m'n'}(x',y')$$

We compute $K_{mn;m'n'}(t)$ for each propagation region in the basis of the unperturbed eigenmodes of the interferometer; then construct a representation of the perturbed interferometer using matrix multiplication.

FABRY-PEROT INTERFEROMETER



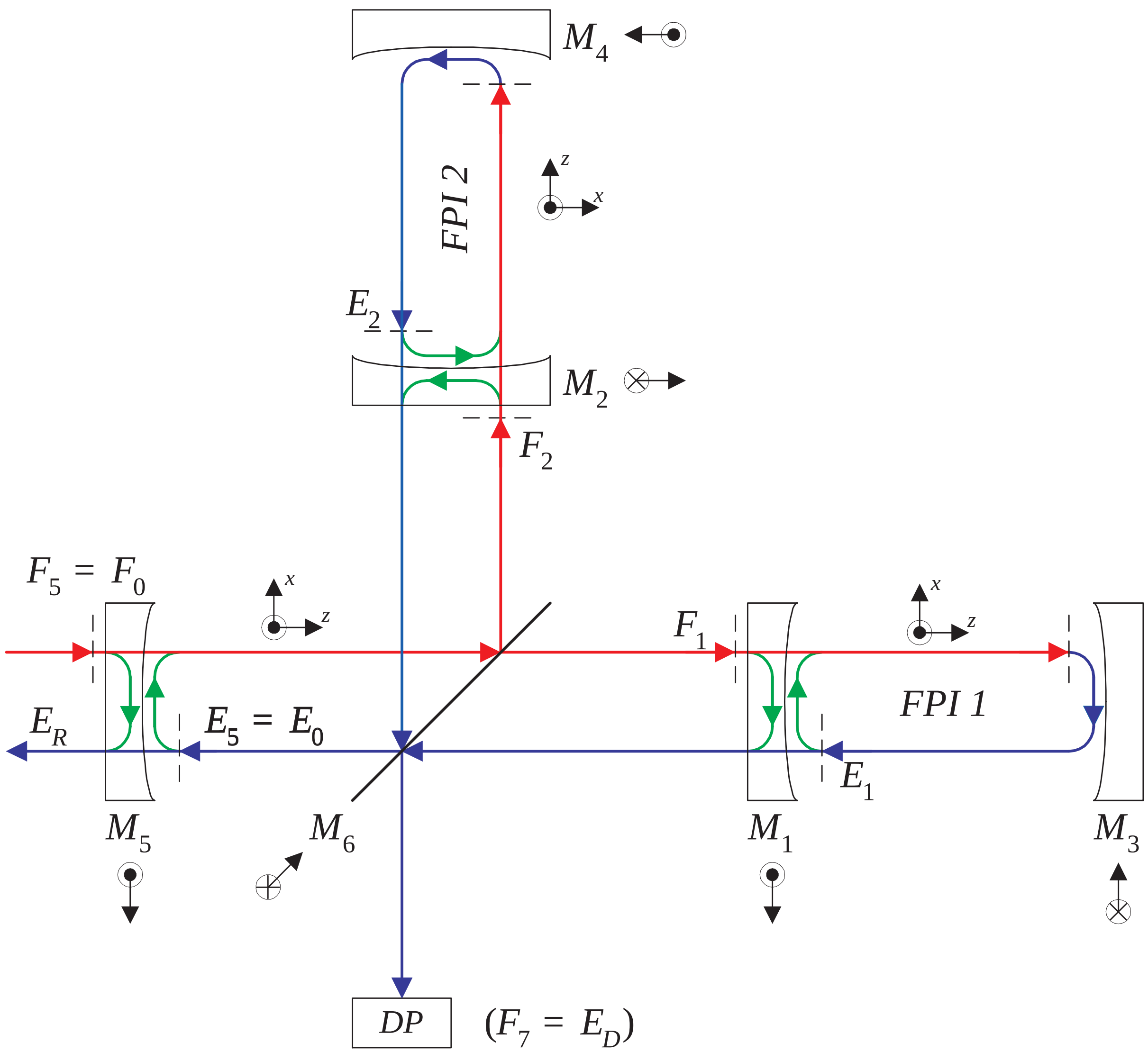
$$A_2 = D_1 = 1 \qquad B_2 = B_1 = L_0$$

$$C_2 = C_1 = 0 \qquad D_2 = A_1 = 1$$

MODEL OF IFO COUPLING

- Choose a *primary* FPI as a reference cavity, defining the "fundamental" unperturbed eigenmodes
- Assume that the reference laser is very nearly mode-matched to the primary FPI
- Propagation around the recycling cavity and through the secondary input mirror directly couples the eigenmodes of the secondary FPI to those of the primary FPI
- Mode-mismatch matrix operators can be included if desired; mirror surface/laser wavefront curvature mismatch provided

IFO COUPLING SCHEMATIC



THERMAL LENSING

- Hello-Vinet model of substrate thermal lensing due to both substrate and coating absorption

- Temperature distribution due to bulk absorption approximated by

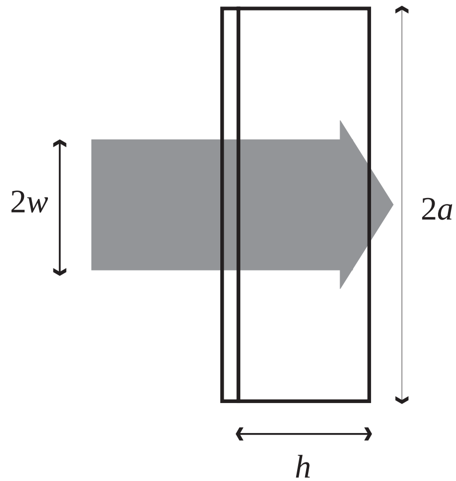
$$T(r) - T(0) = -\frac{\alpha_P P}{4\pi k_T}\left[\gamma + \ln\left(\frac{2r^2}{w^2}\right) + E_1\left(\frac{2r^2}{w^2}\right)\right]$$

- Near $r = 0$, bulk absorption similar to a thin lens with focal length

$$f = \frac{\pi w^2}{\alpha_P h P}\frac{\kappa_T}{dn/dT}$$

- Numerical implementation of astigmatic thermal loading in beam-splitter complete (Hermite-Gauss basis)

HELLO-VINET THERMAL LENS MODEL



Reference: P. Hello and J.-Y. Vinet, J. Phys. France 51, 1267 (1990)

Coating absorption:

$$T_c(r,z) = \frac{P_c}{k_T a} \sum_{k=0}^{\infty} a^2 p_k \left[A_k \cosh\left(\zeta_k \frac{z}{a}\right) + B_k \sinh\left(\zeta_k \frac{z}{a}\right) \right] J_0\left(\zeta_k \frac{r}{a}\right)$$

Substrate absorption:

$$T_s(r,z) = \frac{P_s}{k_T h} \sum_{k=0}^{\infty} \frac{a^2 p_k}{\zeta_k^2} \left[1 - 2\tau A_k \cosh\left(\zeta_k \frac{z}{a}\right) \right] J_0\left(\zeta_k \frac{r}{a}\right)$$

ζ_k**:** Roots of the equation

$$\zeta\, J_1(\zeta) - \tau\, J_0(\zeta) = 0$$

Since $\tau \equiv 4\epsilon T^3 a / k_T = 0.27734$ for fused silica at room temperature,

$$\zeta_k \approxeq (k + 1/4)\,\pi, \quad k \in \{0, 1, 2, \ldots\}$$

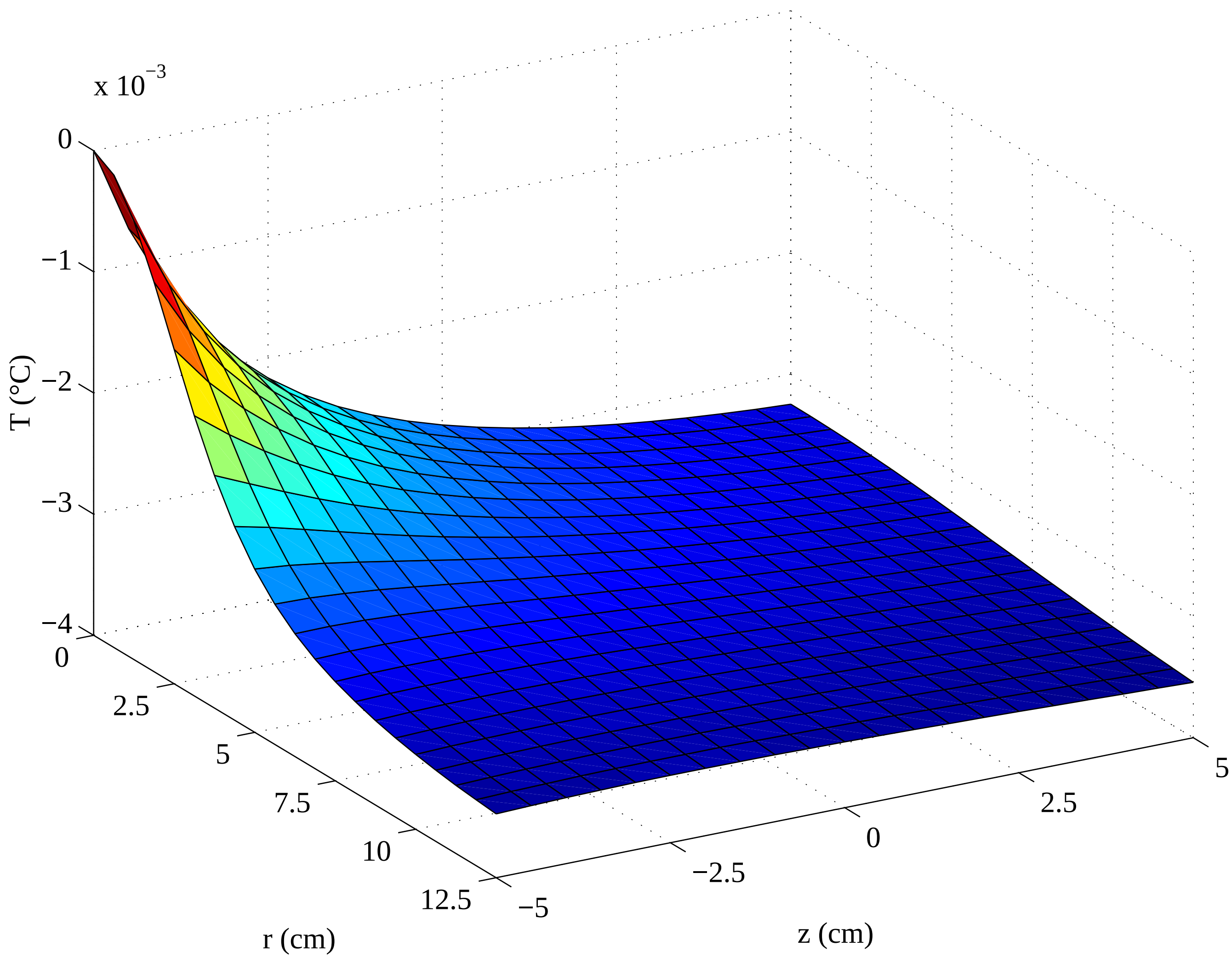
p_k**:** Normalized expansion coefficients

$$u_{00}^2 = \sum_{k=0}^{\infty} p_k\, J_0\left(\zeta_k \frac{r}{a}\right)$$

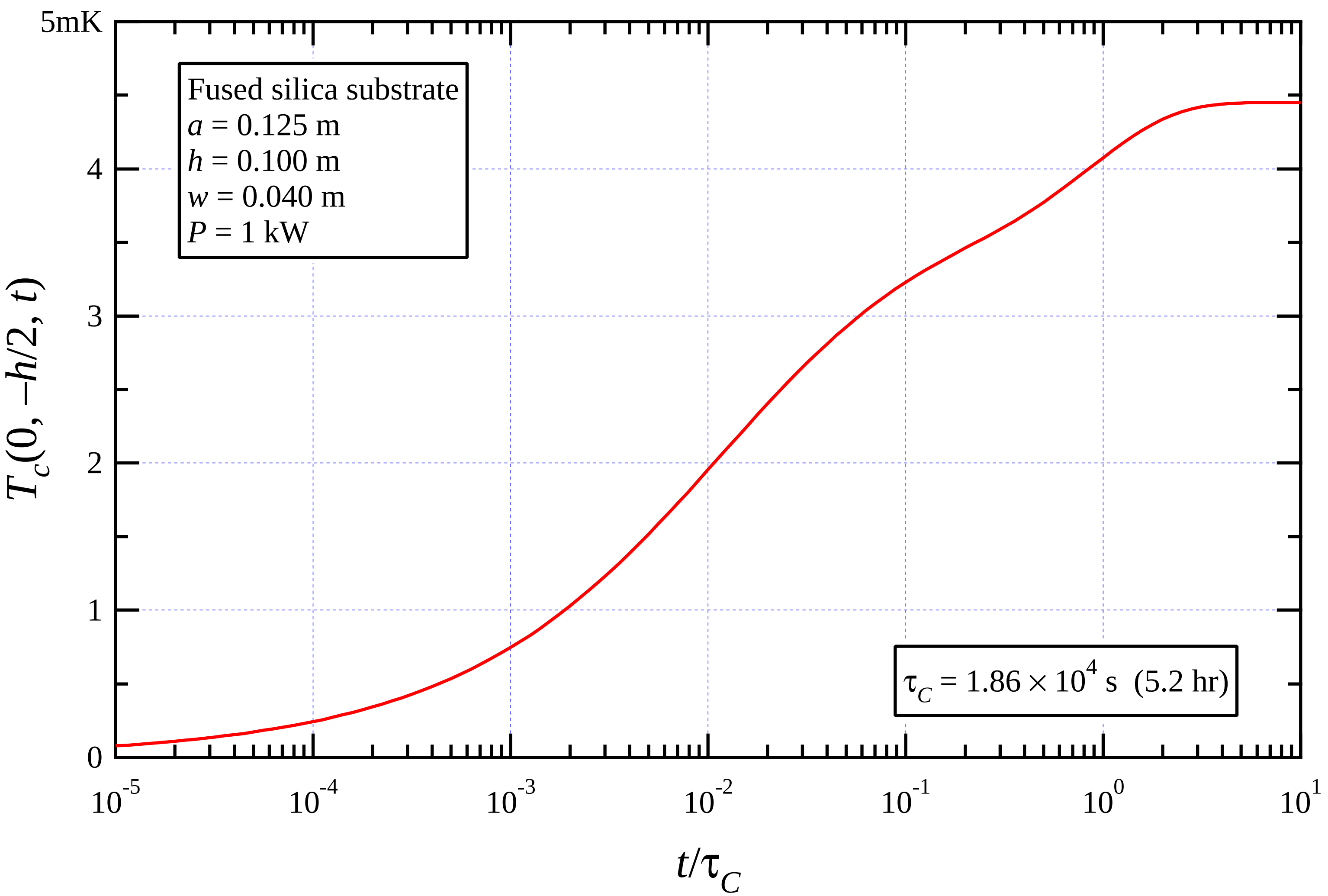
Since $(w/a)^2 \ll 1$,

$$p_k \approxeq \frac{P}{\pi a^2} \frac{\zeta_k^2}{\left(\zeta_k^2 + \tau^2\right) J_0^2(\zeta_k)} e^{-(\zeta_k w/a)^2/8}$$

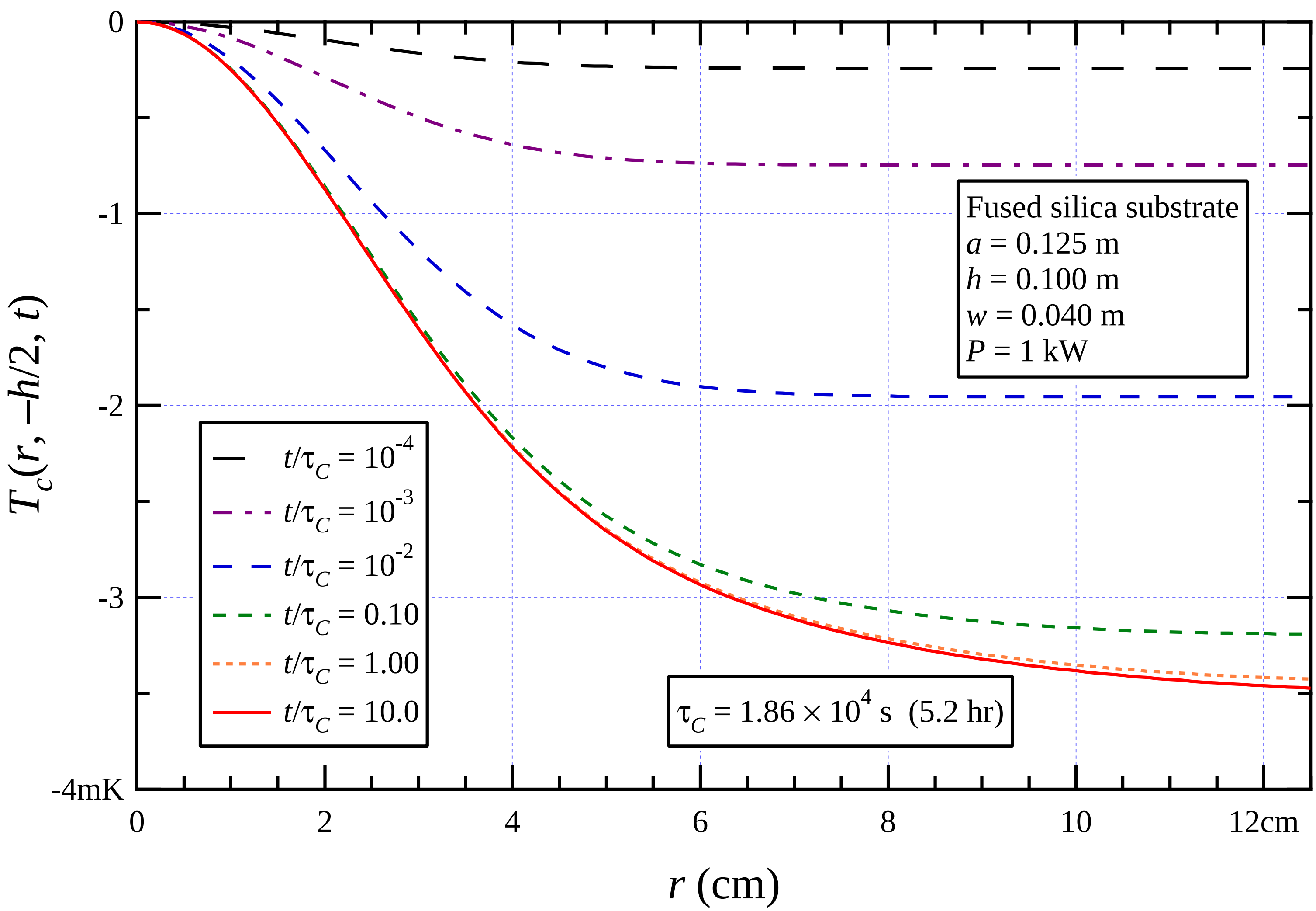
$T_c(r, z)$ FROM COATING ABSORPTION



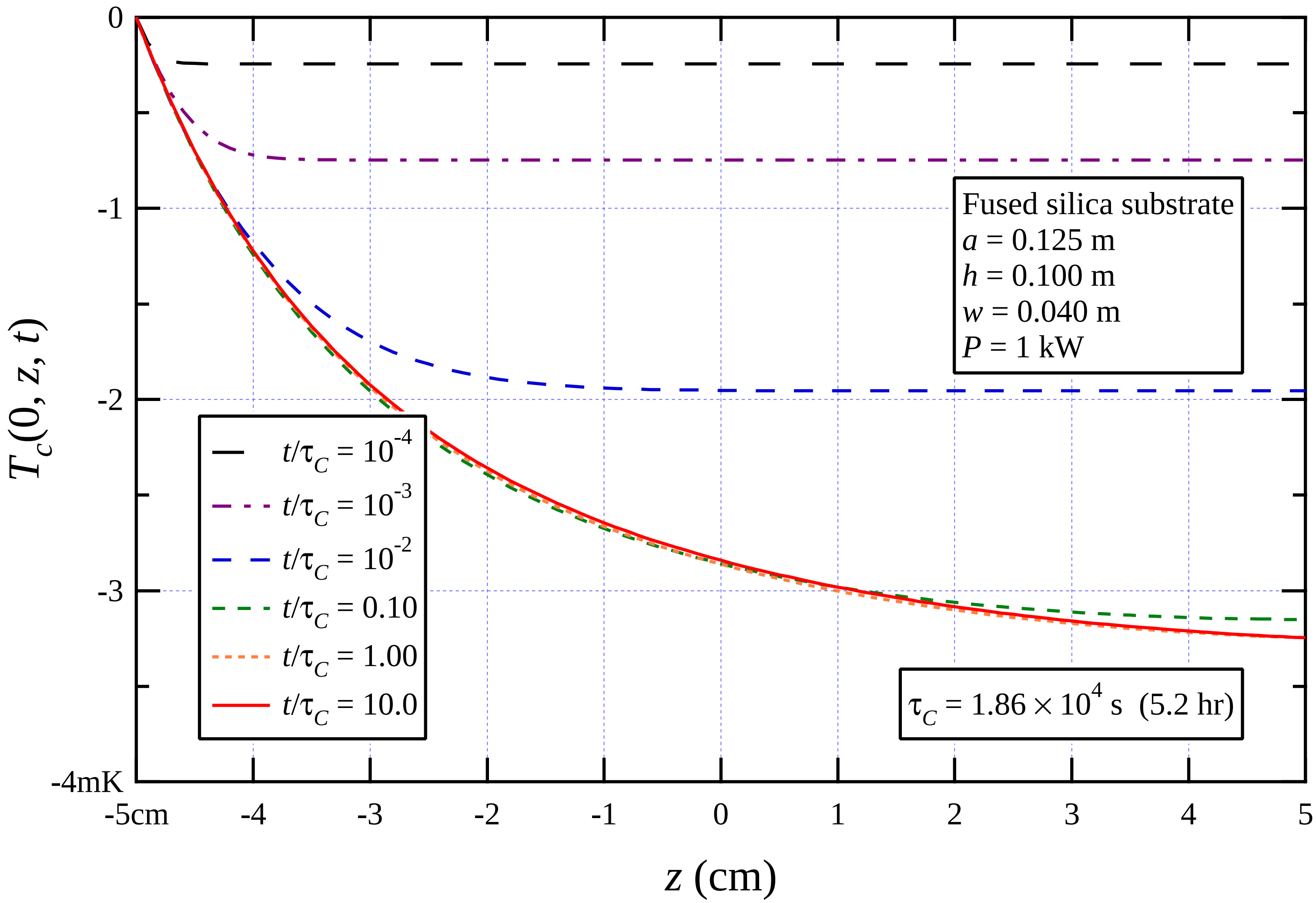
$T_C(0, -h/2, t)$ FROM COATING ABSORPTION



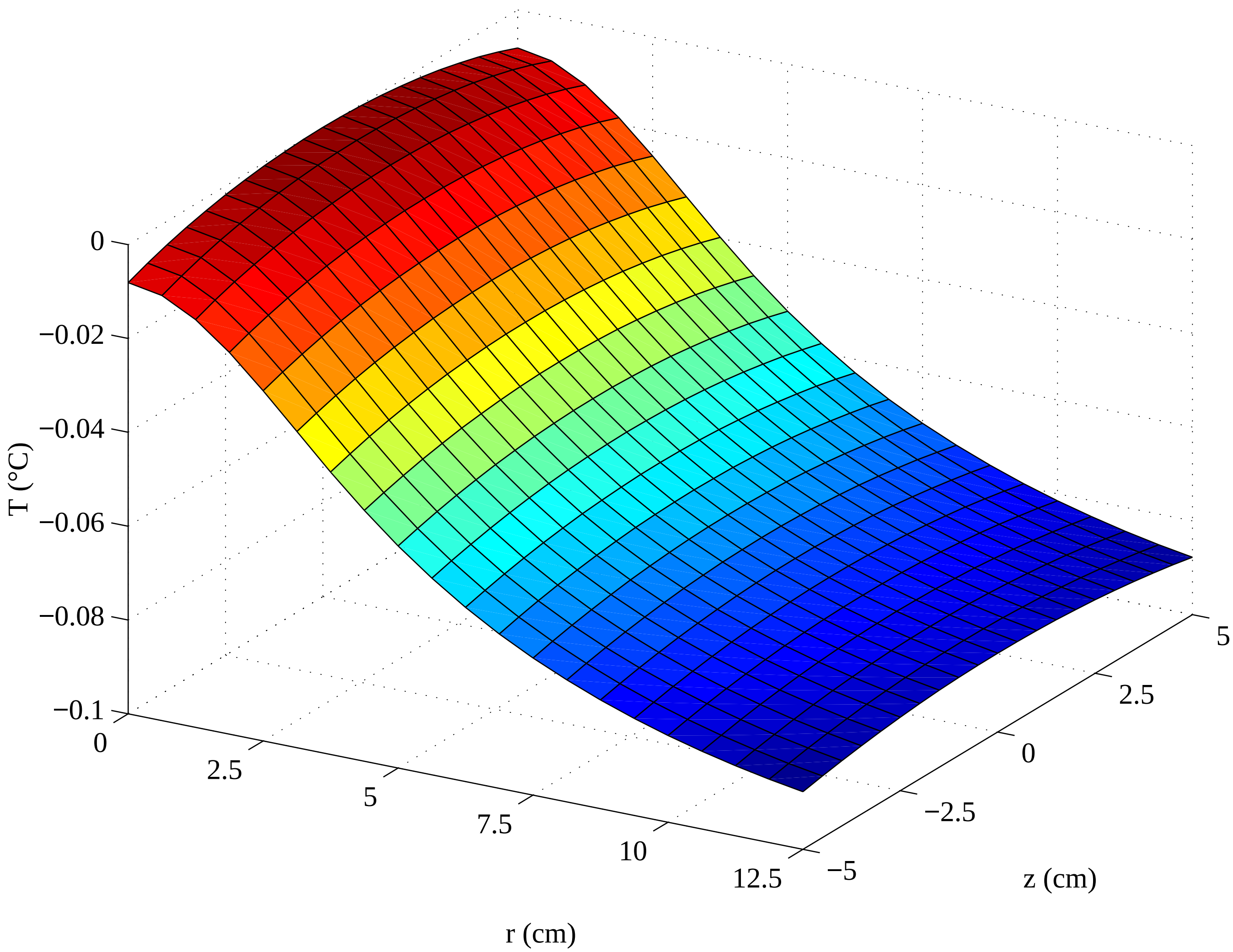
$T_C(r, -h/2, t)$ FROM COATING ABSORPTION



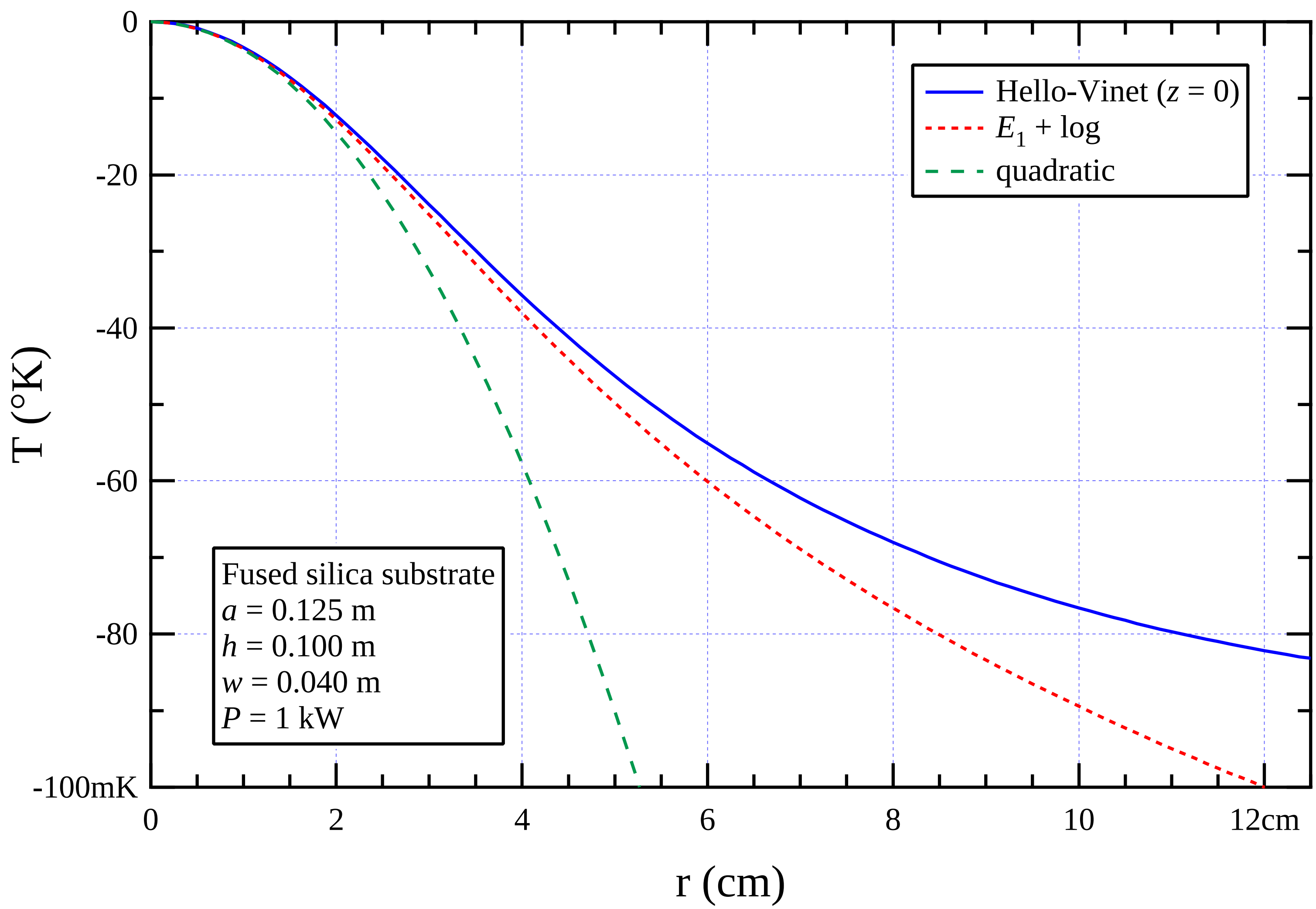
$T_C(0, z, t)$ FROM COATING ABSORPTION



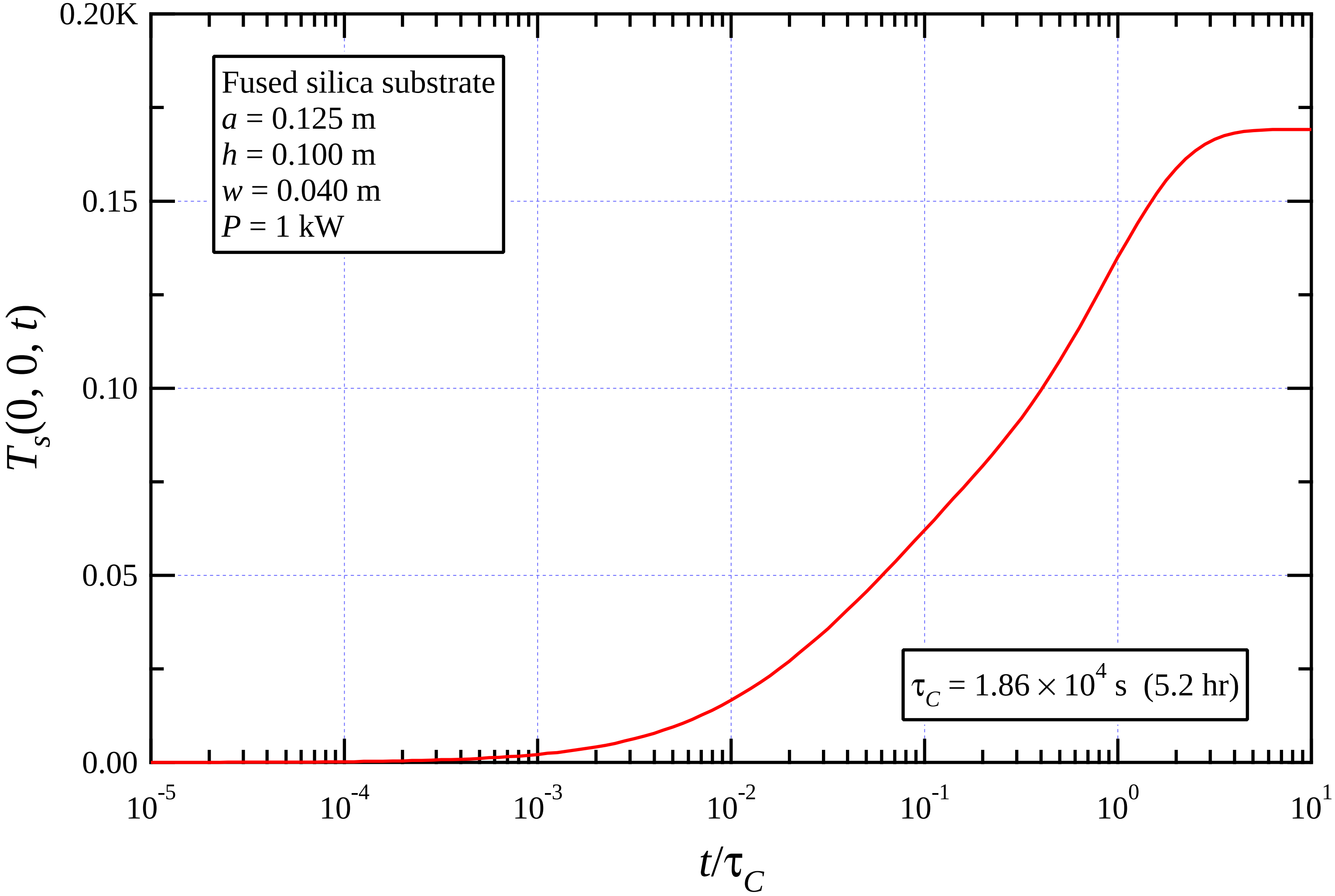
$T_S(r, z)$ FROM SUBSTRATE ABSORPTION



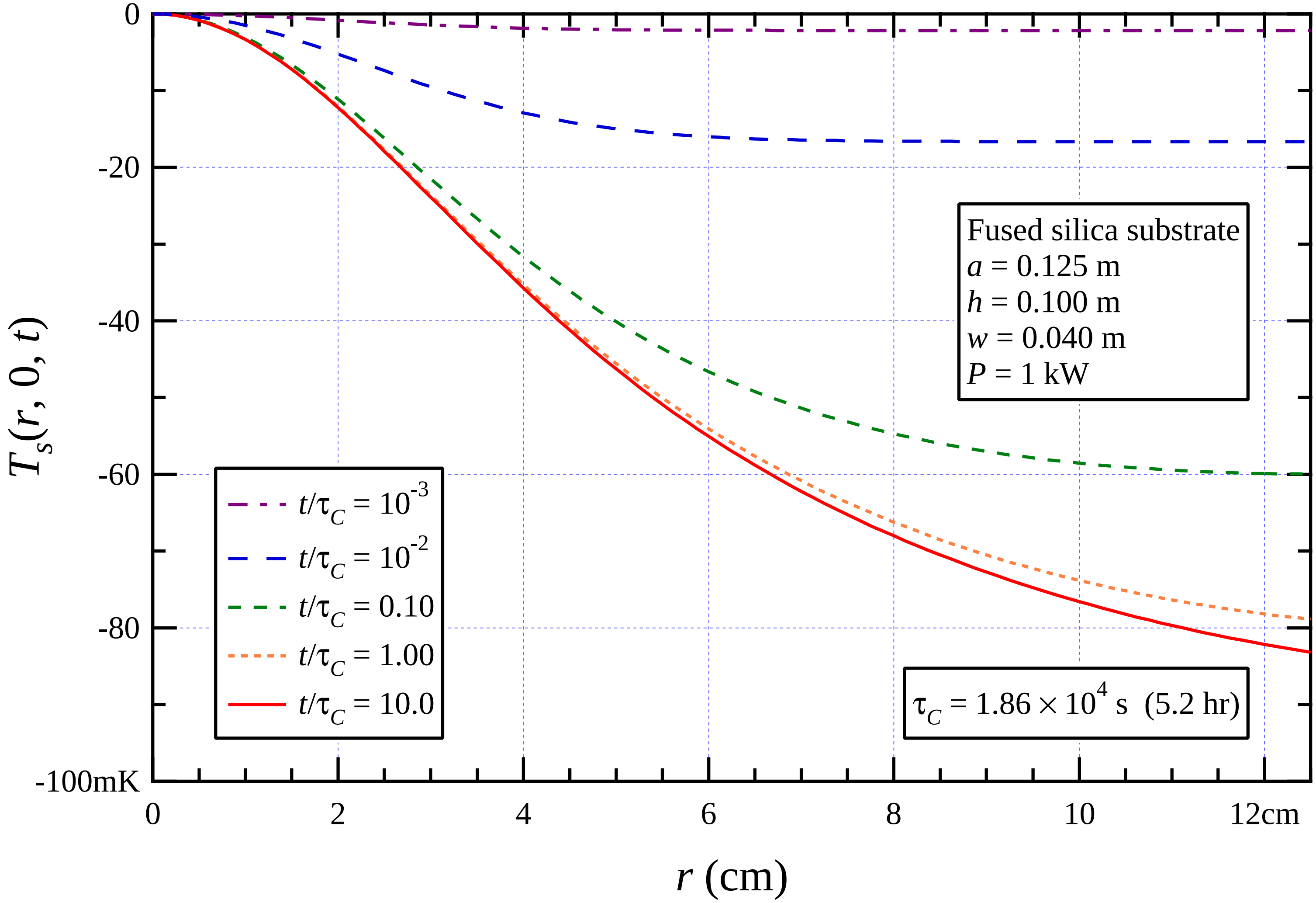
APPROXIMATE $T_S(r,0)$ (SUBSTRATE ABSORPTION)



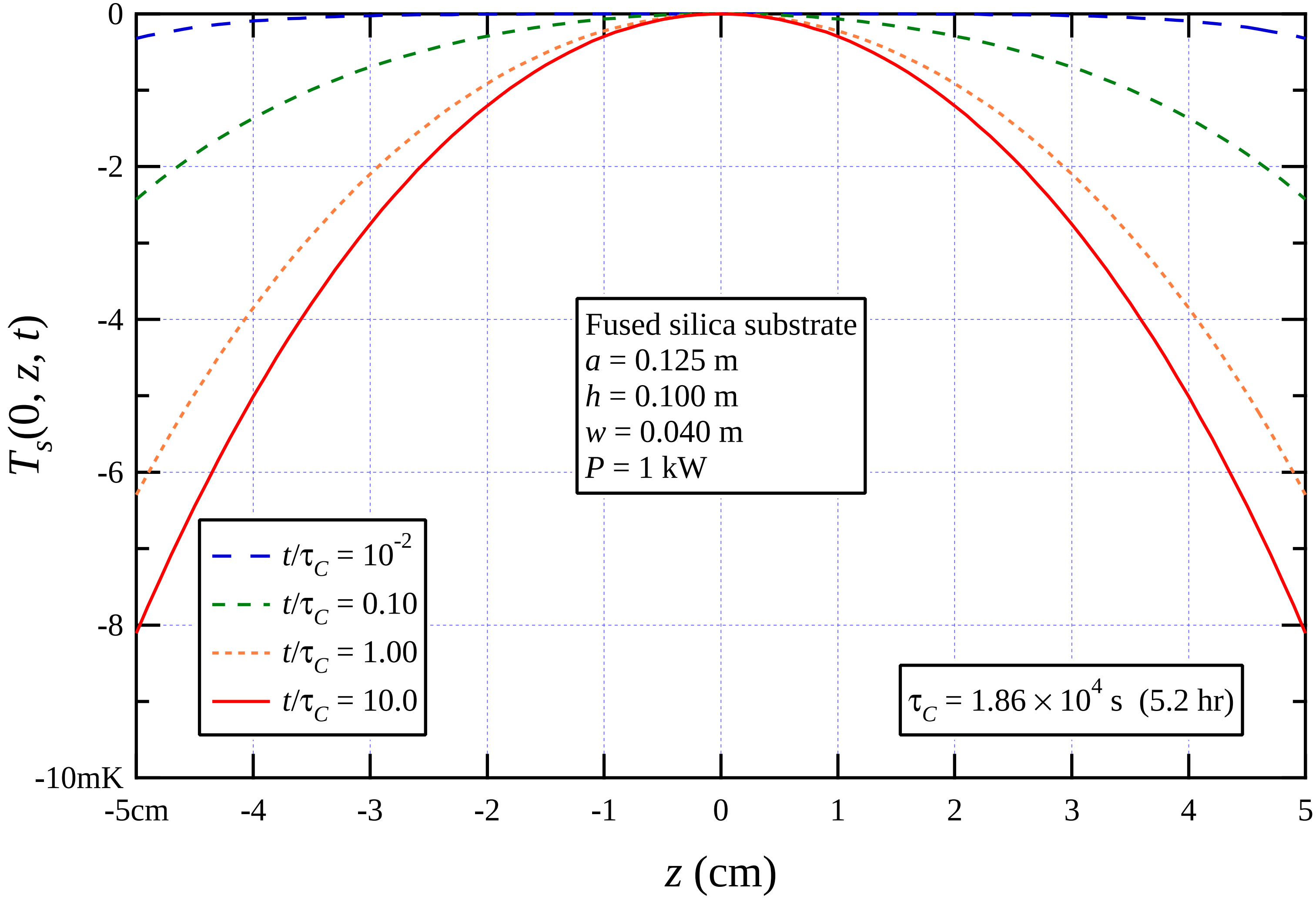
$T_S(0, 0, t)$ FROM SUBSTRATE ABSORPTION



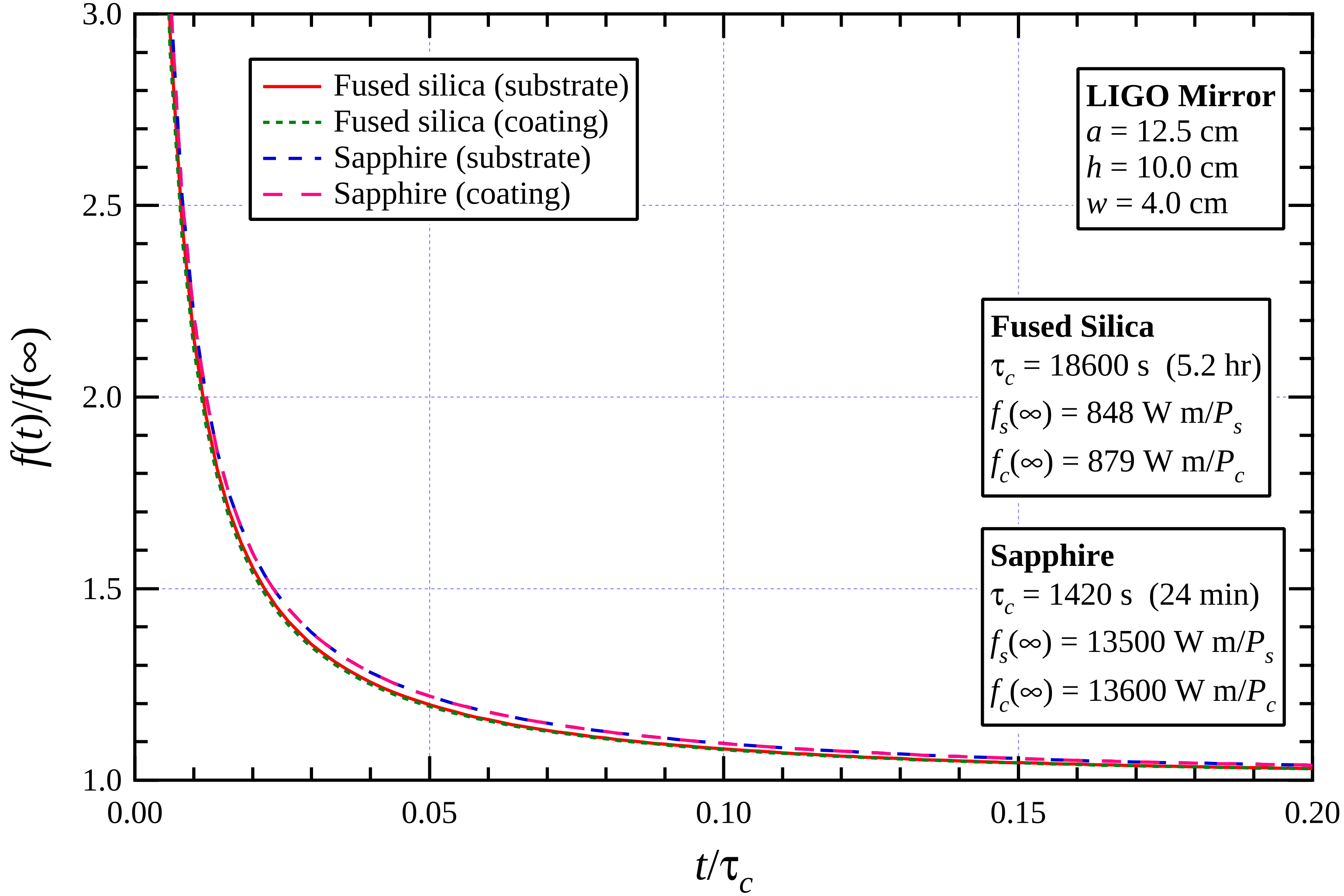
$T_s(r, 0, t)$ FROM SUBSTRATE ABSORPTION



$T_S(0, z, t)$ FROM SUBSTRATE ABSORPTION



THERMAL FOCAL LENGTH



THERMAL LENS OPERATOR

The propagation phase perturbation due to the OPD is

$$\phi(r) = \frac{2\pi}{\lambda_0} \frac{dn}{dT} \int_{-h/2}^{h/2} dz\, T(r,z)$$

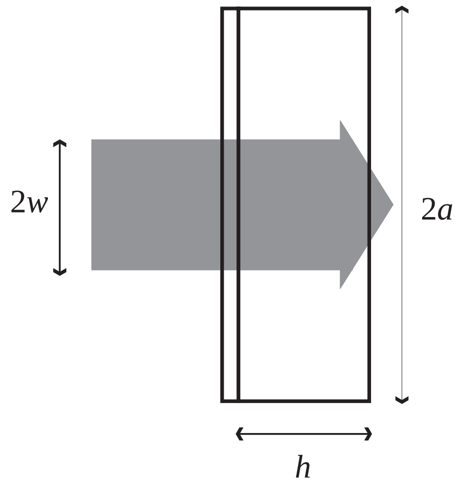
where $T(r,z)$ is the *linear* sum of contributions from heating due to absorption in both coatings (HR and AR) and the substrate.

Matrix elements of the thermal lens operator:

$$\begin{aligned} \Phi_{m'n';mn} &= \iint_{-\infty}^{\infty} dx dy\, u^{\dagger}_{m'n'}(x,y)\, u_{mn}(x,y)\, e^{i\phi(r)} \\ &\equiv \exp\left[i \iint_{-\infty}^{\infty} dx dy\, u^{\dagger}_{m'n'}(x,y)\, u_{mn}(x,y)\, \phi(r) \right] \end{aligned}$$

Since $\phi(r) \propto r^2$, TEM_{00} is coupled to both TEM_{20} and TEM_{02}.

HELLO-VINET THERMOELASTIC SURFACE DEFORMATION



Reference: P. Hello and J.-Y. Vinet, J. Phys. France 51, 2243 (1990)

$$\gamma_k \equiv \zeta_k h/2a$$

Coating absorption (neglect Saint Venant correction):

$$u_c\left(r,-\frac{h}{2}\right) = P_c\frac{\alpha_T}{k_T}(1+\nu)\sum_{k=0}^{\infty}\frac{a^2 p_k}{\zeta_k}\left[A_k\cosh(\gamma_k)+B_k\sinh(\gamma_k)\right]J_0\left(\zeta_k\frac{r}{a}\right)$$

Substrate absorption:

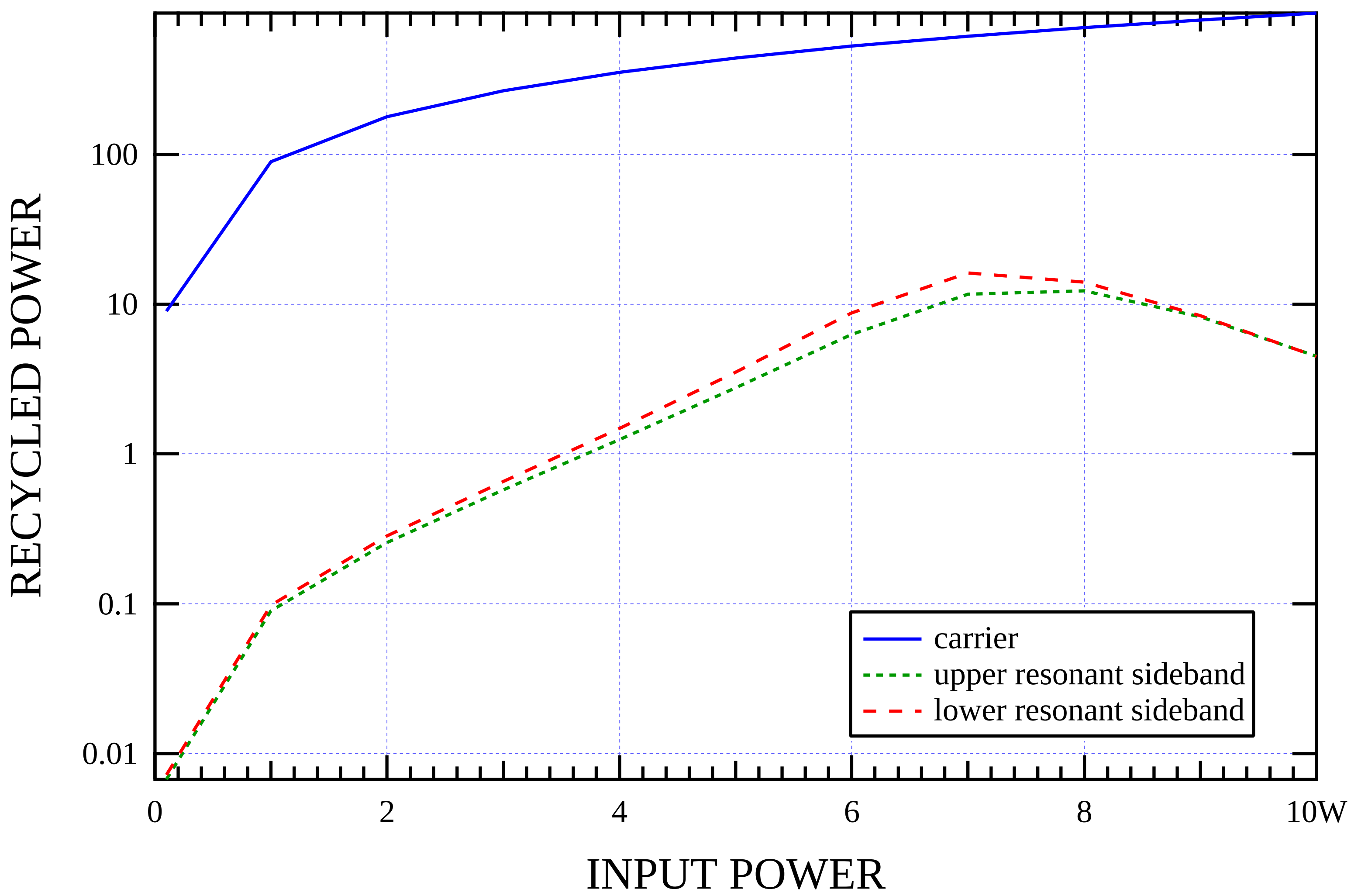
$$u_s\left(r,-\frac{h}{2}\right) = P_s\frac{\alpha_T}{k_T}(1+\nu)\sum_{k=0}^{\infty}\frac{a^2 p_k}{\zeta_k^2}\tau A_k\frac{\sinh(\gamma_k)}{\gamma_k}J_0\left(\zeta_k\frac{r}{a}\right)$$

RESONATOR LENGTH PSEUDOLOCKING

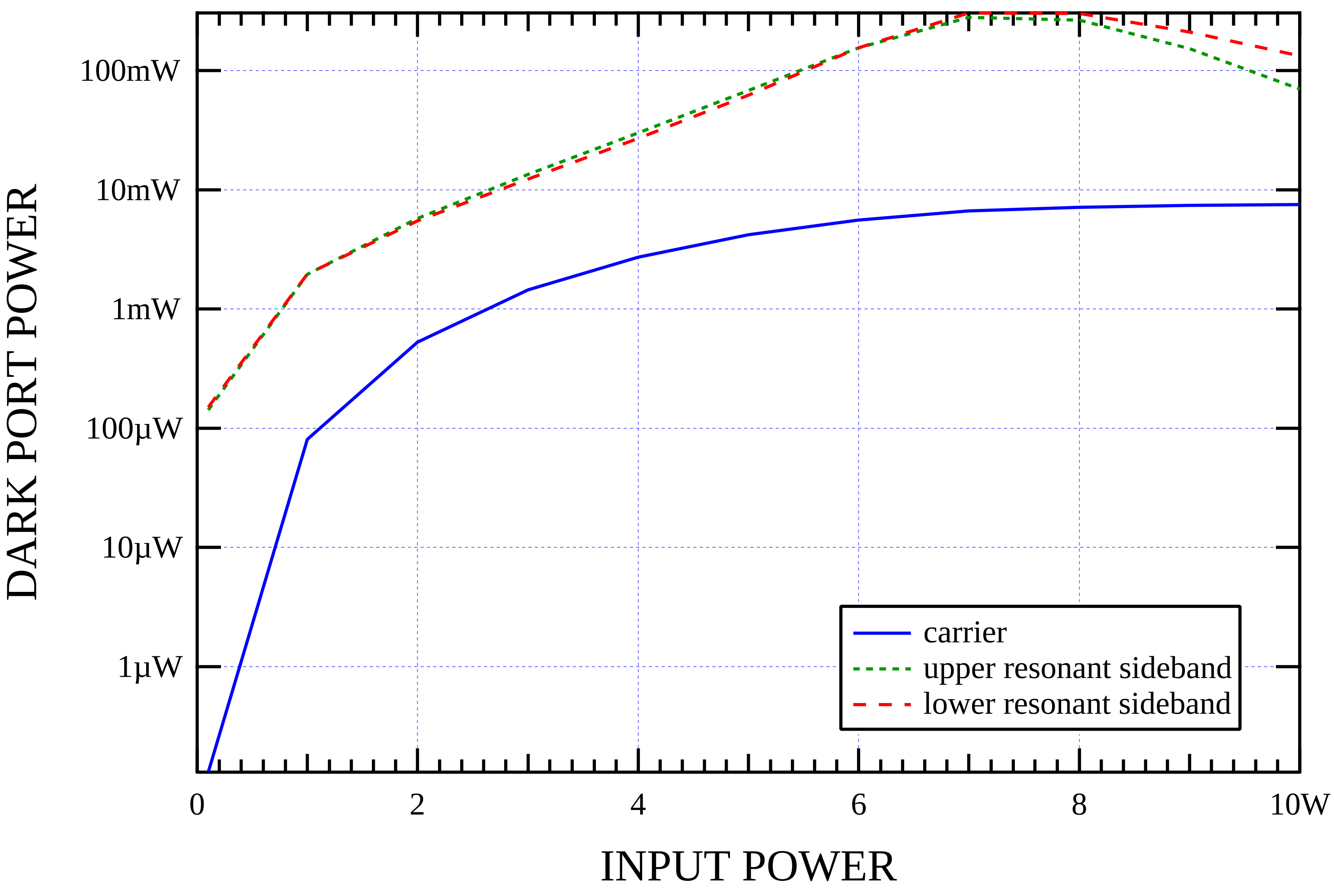
Self-contained simulations: implicit four-stage *pseudolocker*

1. *FPI* stage adjusts the positions of the FPI ITMs to maximize round-trip carrier TEM_{00} enhancement.
2. *Dark Port* stage adjusts the beamsplitter position so that the amplitude of the carrier TEM_{00} mode is minimized at the dark port.
3. *Power Recycling* stage adjusts the position of the PR mirror to maximize carrier TEM_{00} enhancement.
4. *Signal Recycling* stage adjusts the position of the SR mirror to optimize carrier TEM_{00} phase at the SR.

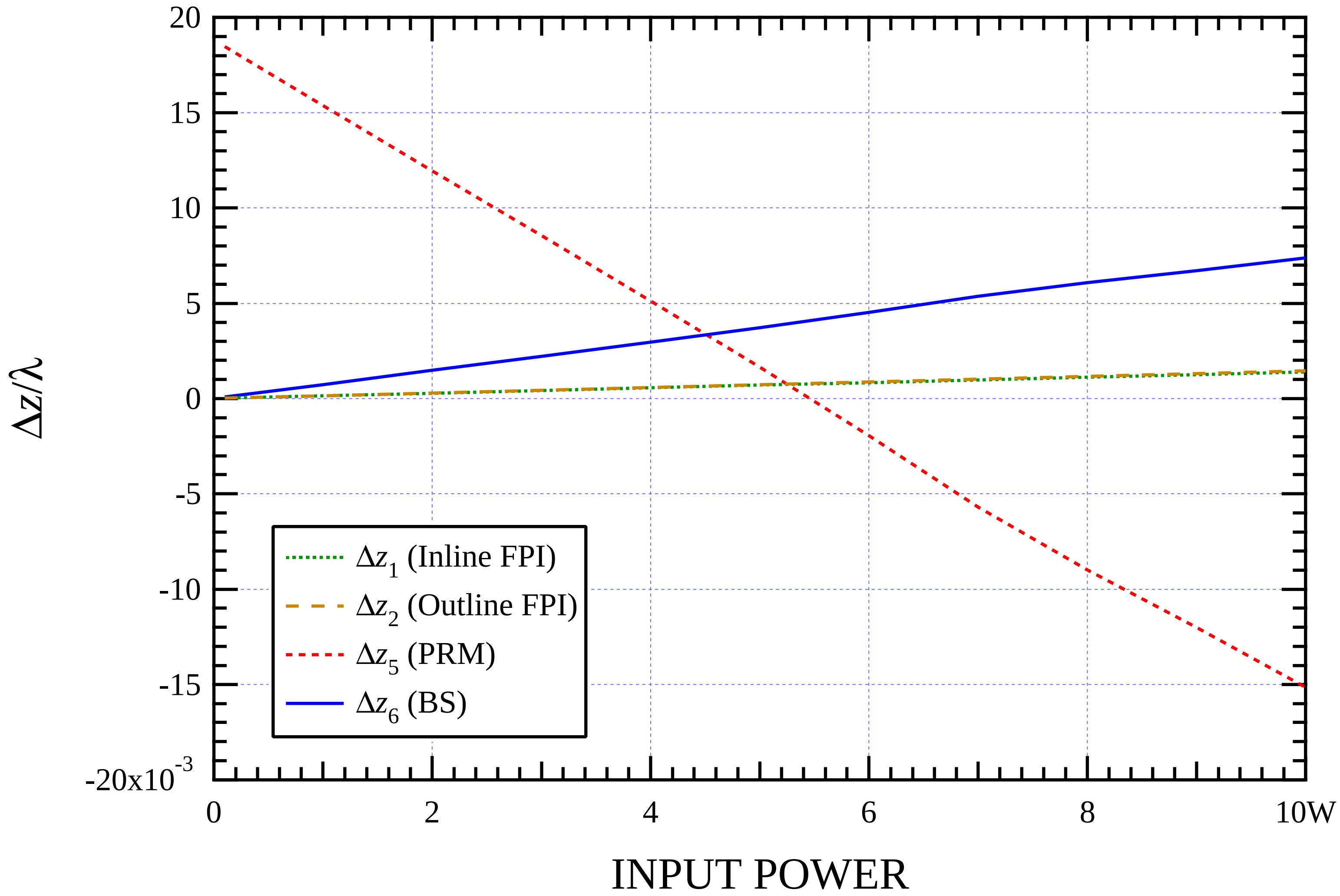
LIGO I RECYCLED POWER



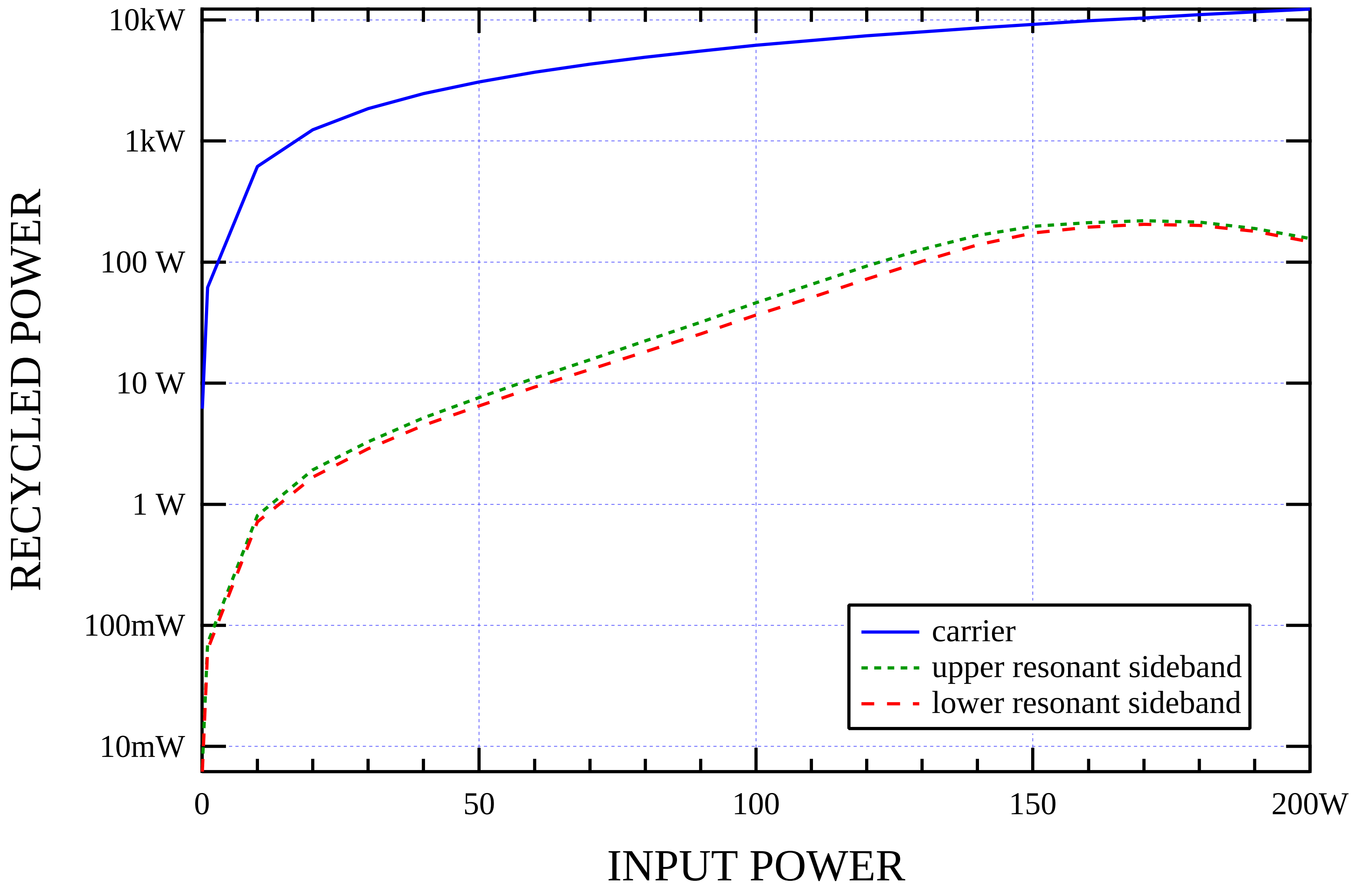
LIGO I DARK PORT OUTPUT POWER



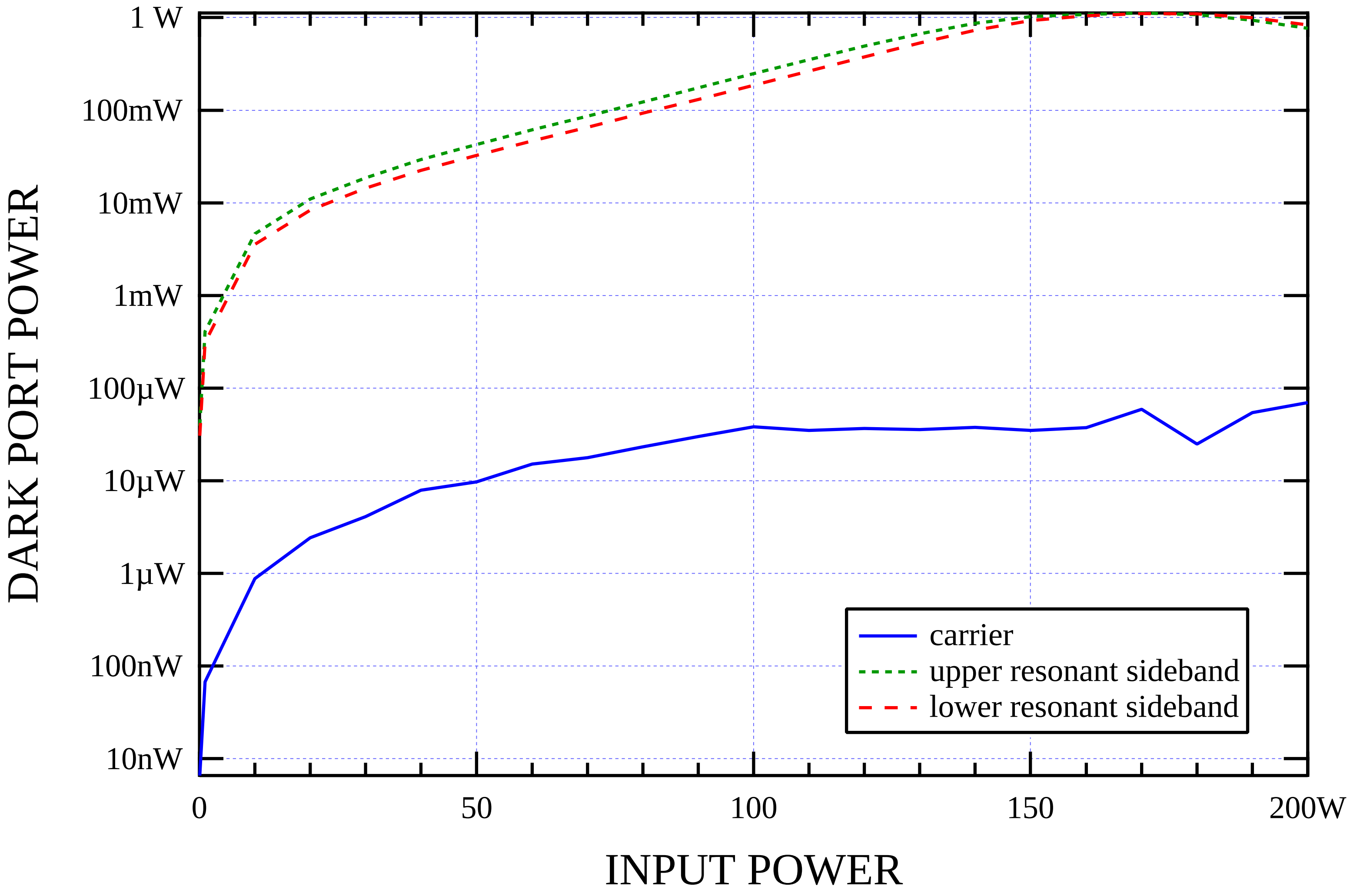
LIGO I PSEUDOLOCKER OUTPUT



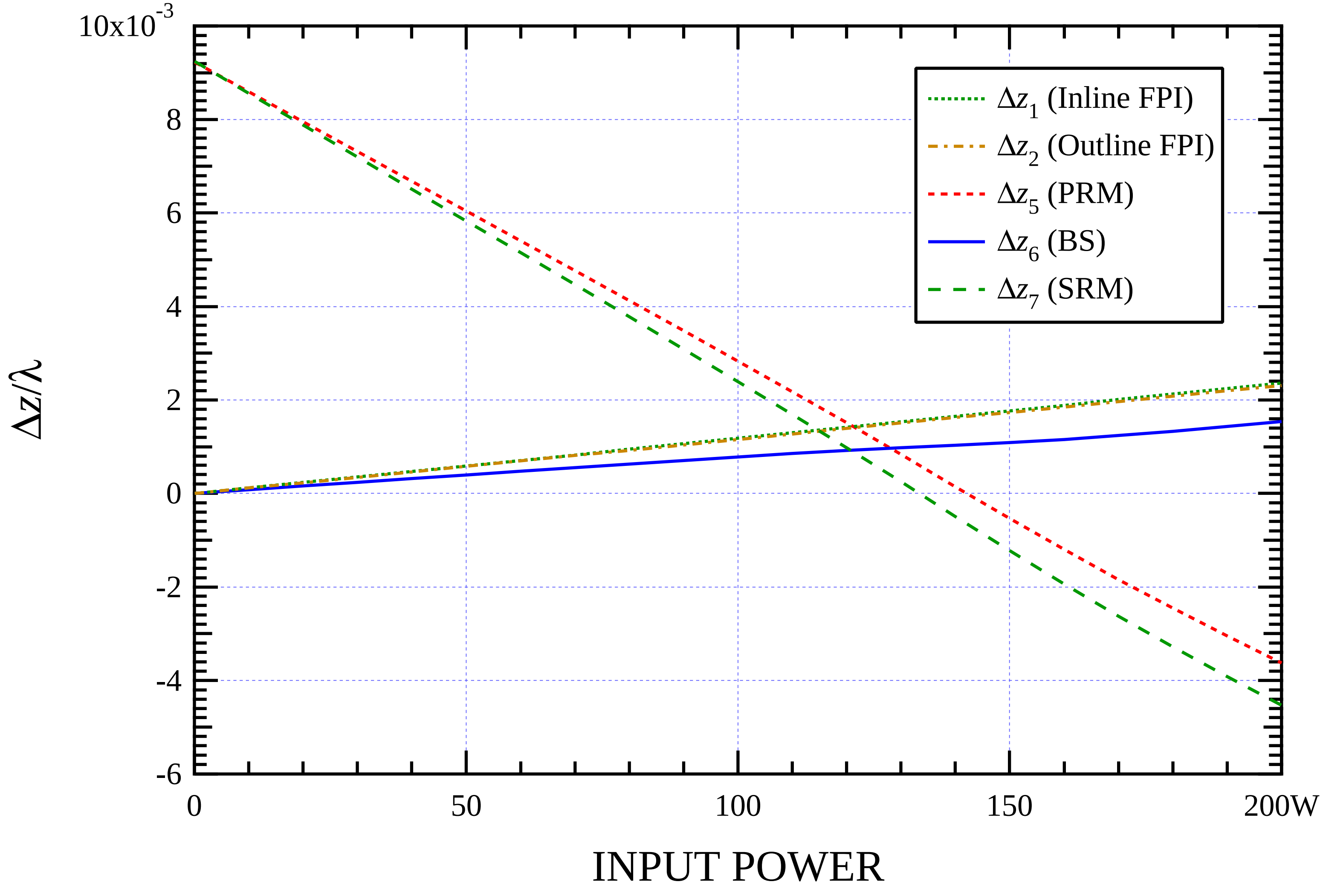
LIGO II RECYCLED POWER



LIGO II DARK PORT OUTPUT POWER



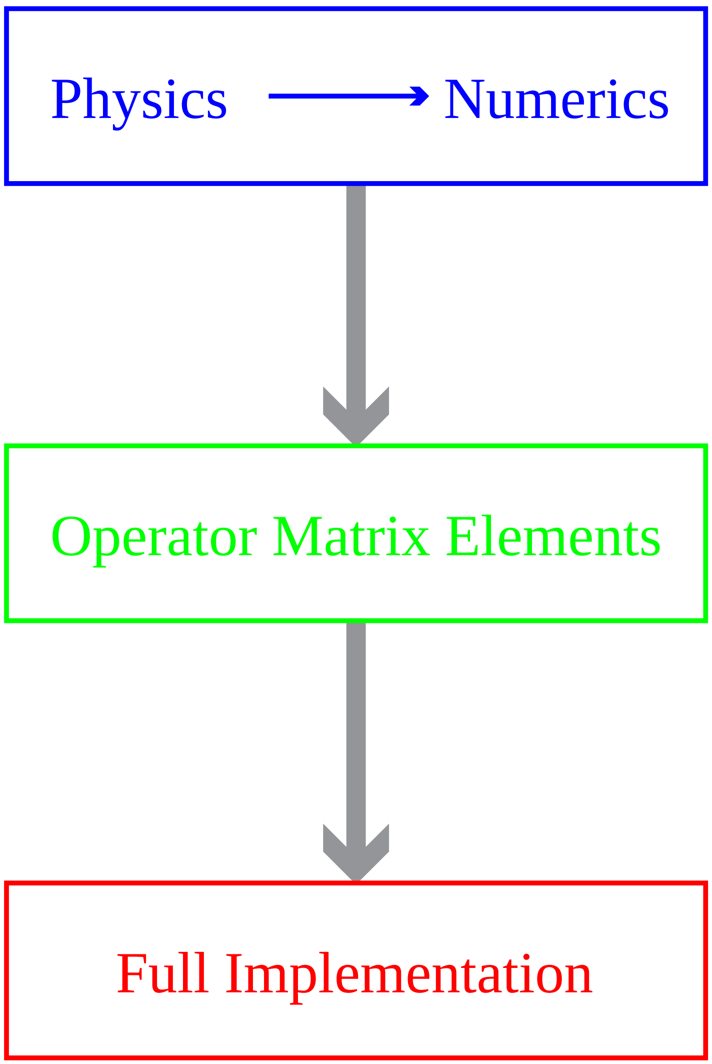
LIGO II PSEUDOLOCKER OUTPUT



MELODY/MATLAB FEATURES

- Simple object-oriented architecture in MATLAB
- Flexible modulation and resonance schemes
- Hello-Vinet mirror thermal lens and surface deformation
- Aperture diffraction for arbitrary number of spatial modes
- Pseudolockers for LIGO I/II
- Astigmatic beamsplitter thermal lens
- Mirror/field curvature mismatch
- Precomputation of all matrix operators available

MELODY FEATURE LIFE-CYCLE



- Transient thermal loading
- Static thermal compensation
- Mirror misalignment
- Static thermal loading
- Static thermal deformation

NEW FEATURE PRIORITIES

1. Numerical calculation of beamsplitter thermoelastic surface deformation $\longrightarrow 45^\circ$ incidence
2. Correct model of thermal compensation
3. Demodulation routine for detector class
4. Faster engine
5. Transient thermal loading
6. Mirror misalignment operators

IMPLEMENTATION

- Use object-oriented programming in MATLAB: primitive classes, encapsulation, function/operator overloading, and inheritance
- Define classes for mirrors, Fabry-Perot interferometers, electric fields (Hermite-Gauss, RF-modulated), and detectors ($\{x, y\}$ geometry)
- Enclose classes representing simpler entities (mirrors, beamsplitters, laser fields) in classes representing interferometers
- Design simple class interfaces allowing calculations and simulations to be driven by MATLAB scripts
- Automatic translation to C++ available if performance is an issue

User-defined
driver scripts

Melody architecture
and classes (@mirror, ...)

“*Cui ci sono dei mostri.*”

SCRIPT-LEVEL FEATURES

- Input powers, modulation frequencies and depths
- All mirror parameters (e.g., thermal constants, orientation and micro-position)
- All interferometer cavity lengths
- Power/signal recycling
- Iteration and solution methods
- Graphics, object storage(!), post-processing
- Full interactive MATLAB functionality

TWO-PHASE THERMAL/TEMPORAL SIMULATIONS

Characteristic time: $t_c = \rho C a^2 / k_T \approx 5$ h for fused silica ($a = 0.125$ m)

THERMAL (Script-driven)

1. Run thermal relaxation code, including power-dependent optimizations (e.g., modulation depths, SRM reflectivity)
2. SAVE ligo object after stability is reached for each power level

TEMPORAL (Script-driven, SIMULINK)

1. LOAD ligo object for a specified input power
2. Perturb mirrors and simulate temporal response

SUBSET OF CLASSES

`laser_field` Stores all spatial components for all operating sidebands, and the frequencies of those sidebands.

`mirror` Maintains all perturbation matrices (e.g., thermal and angular); encapsulates mirror parameters, two laser_field objects, detectors.

`beamsplitter` Special case of mirror for 45° beamsplitter; uses numerical temperature distribution.

`detector` Demodulation detector array; almost complete.

`fpi` Fabry-Perot Interferometer

`ligo` LIGO I/II, GEO 600 Interferometers

LASER_FIELD OBJECT DATA

```
basis: Hermite-Gauss

[f\_0, f\_1, -f\_1, f\_2, -f\_2] = [0,  2.3971e+001, -2.3971e+001, 3.5956e+001, -3.5956e+001]

TEM_00   -7.36e-01 -2.45e-02i  -7.48e-02 -5.45e-04i  -7.48e-02 -4.79e-04i  -5.35e-02 -2.04e-03i -5.35e-02 -7.46e-04i
TEM_10             0                     0                     0                     0                    0
TEM_01             0                     0                     0                     0                    0
TEM_20   -1.64e-04 +1.86e-04i   8.15e-03 -9.54e-03i   7.54e-03 -9.98e-03i  -6.69e-07 +6.32e-06i -9.64e-08 +6.47e-06i
TEM_11             0                     0                     0                     0                    0
TEM_02   -1.64e-04 +1.86e-04i   8.15e-03 -9.54e-03i   7.54e-03 -9.98e-03i  -6.69e-07 +6.32e-06i -9.64e-08 +6.47e-06i
```

- laser_field class consists of data fields (*members*) and routines which operate on those fields

- Routines fall into two broad categories:

  procedures which alter the internal state of the object but do not return results (e.g., object update procedures)

  functions which return results but do not alter the internal state of the object (e.g., overloaded arithmetic operators)

Define the propagation vector

$$k = k_0 + \Delta k_q,$$

where $\Delta k_q / k_0 \ll 1$, $\omega_0 \equiv k_0 c$, and $\Delta\omega_q \equiv \Delta k_q c$. Write the time-dependent length as

$$L(t) \equiv L_0 + \Delta L(t),$$

where $2k_0 L_0 - \varphi_{00} = 2N\pi$ and $\Delta L(t) \approx \lambda = 2\pi / k_0$. Then

$$\begin{aligned} e^{i[2kL(t)-\varphi_{00}]} &= e^{i(2k_0L_0-\varphi_{00})} e^{i[2k_0\Delta L(t)]} e^{i\left(2\Delta\omega_q L_0/c\right)} e^{i\left[2\Delta k_q \Delta L(t)\right]} \\ &= e^{i\left[2k_0\Delta L(t)+\Delta\omega_q\tau_0\right]} \end{aligned}$$

Include $\Delta L(t)$ in mirror class; implement $\Delta\omega_q\tau_0$ as a diagonal propagation matrix.

FPI OBJECT UPDATE PROCEDURE

```
% Get the total field propagating away from the
% vacuum-coating interface of m_1, and then
% propagate that field to the vacuum-coating
% interface of m_2. This is the new 'front
% field' of m_2.
e_1_r = get_field(m_1, 'front');
e_2 = fp.gouy_prop * e_1_r * fp.kz_prop;
set_field(m_2, e_2, 'front');

% Get the total field propagating away from the
% vacuum-coating interface of m_2, and then
% propagate that field to the vacuum-coating
% interface of m_1. This is the new 'front
% field' of m_1.
e_2_r = get_field(m_2, 'front');
e_1 = fp.gouy_prop * e_2_r * fp.kz_prop;
set_field(m_1, e_1, 'front');
```

LASER_FIELD MTIMES FUNCTION

```
function e_3 = mtimes(e_1, e_2)
%
...
%
if isa(e_1, 'laser_field') & ~isa(e_2, 'laser_field')
% Initialize the structure e_3 with the same basis and sidebands
% as e_1, and multiply (matrix, using *) the elements of the
% matrix e_2 by the components of e_1.
   e_3.basis = e_1.basis;
   e_3.sideband = e_1.sideband;
   e_3.component = e_1.component*e_2;
elseif ~isa(e_1, 'laser_field') & isa(e_2, 'laser_field')
% Initialize the structure e_3 with the same basis and sidebands
% as e_2, and multiply (matrix, using *) the components of
% e_2 by the elements of the matrix e_1.
   e_3.basis = e_2.basis;
   e_3.sideband = e_2.sideband;
   e_3.component = e_1*e_2.component;
else
   error('Matrix multiplication of two laser_field objects is not allowed.');
end

% Create a new laser_field object from the struct e_3.
e_3 = class(e_3, 'laser_field');
```

MATLAB/OOP REFERENCES

- Duane Hanselman and Bruce Littlefield, **Mastering MATLAB 5: A Comprehensive Tutorial and Reference** (Prentice-Hall, 1998); ISBN 0-13-858366-8
- Bertrand Meyer, **Object-Oriented Software Construction**, Second Edition (Prentice-Hall, 1997); ISBN 0-13-629155-4
- Paul F. Dubois, **Object Technology for Scientific Computing: Object-Oriented Numerical Software in Eiffel and C** (Prentice-Hall, 1997); ISBN 0-13-257808-X
- John J. Barton and Lee R. Nackman, **Scientific and Engineering C++** (Addison-Wesley, 1995); ISBN 0-201-53393-6