Efficient Automatic Differentiation of Matrix Functions

Peder A. Olsen, Steven J. Rennie, and Vaibhava Goel
IBM T.J. Watson Research Center

Automatic Differentiation 2012
Fort Collins, CO

July 25, 2012

How can we compute the derivative?

Numerical Methods that do numeric differentiation by formulas like the finite difference

$$f'(x) \approx \frac{f(x+h) - f(x)}{h}$$

These methods are simple to program, but loose half of all significant digits.

Symbolic Symbolic differentiation gives the symbolic representation of the derivative without saying how best to implement the derivative. In general symbolic derivatives are as accurate as they come.

Automatic Something between numeric and symbolic differentiation. The derivative implementation is computed from the code for $f(x)$.

- **Finite Difference:**

$$f'(x) \approx \frac{f(x+h) - f(x)}{h}$$

- **Central Difference:**

$$f'(x) \approx \frac{f(x+h) - f(x-h)}{2h}$$

For $h < \epsilon x$, where ϵ is the machine precision, these approximations become zero. Thus h has to be chosen with care.

The error in the finite difference is bounded by

$$\|f''\|_\infty \frac{h}{2} + \frac{2\epsilon\|f\|_\infty}{h}$$

whose minimum is

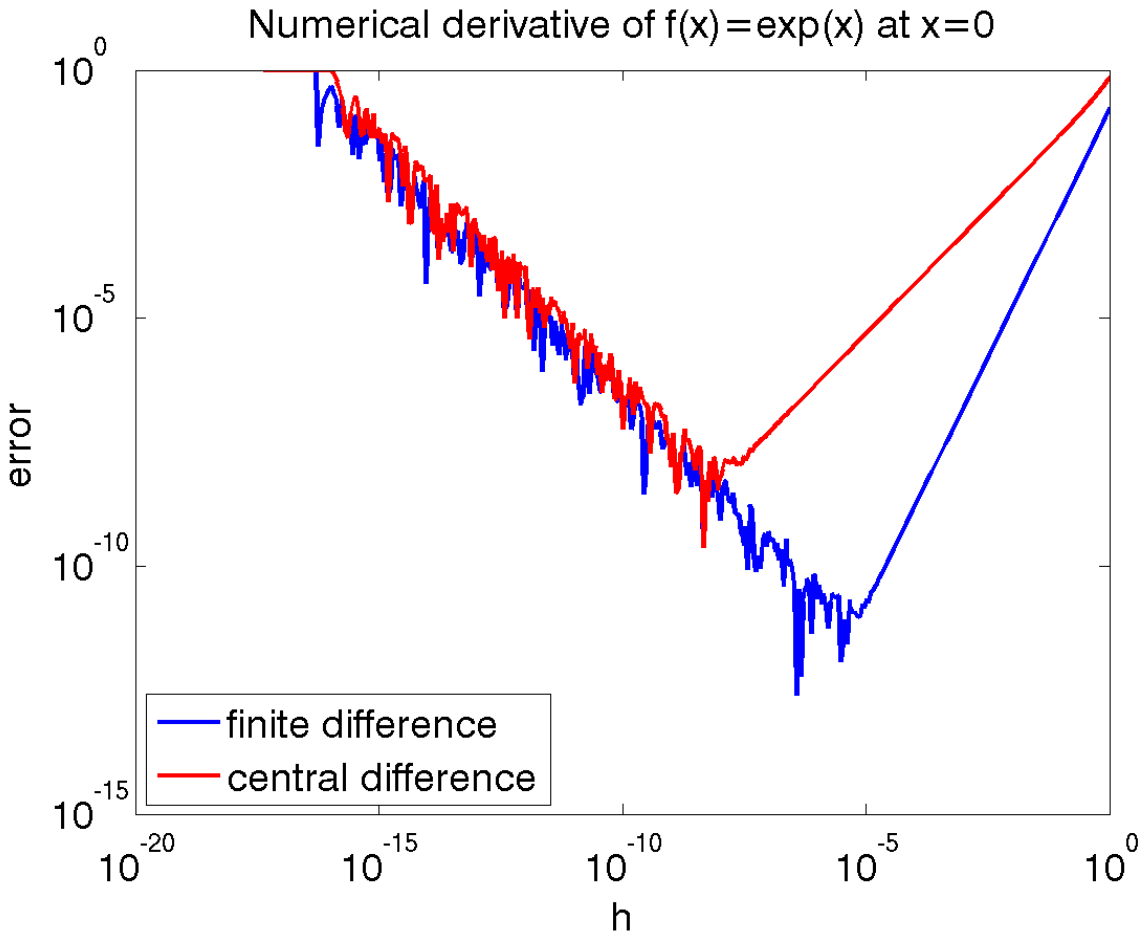
$$2\sqrt{\epsilon\|f\|_\infty\|f''\|_\infty}$$

achieved for

$$h = 2\sqrt{\frac{\epsilon\|f\|_\infty}{\|f''\|_\infty}}.$$

Numerical Differentiation Error

The graph shows the error in approximating the derivative for $f(x) = e^x$ as a function of h.



An imaginary derivative

The error stemming from the subtractive cancellation can be completely removed by using complex numbers.
Let $f : \mathbb{R} \to \mathbb{R}$ then

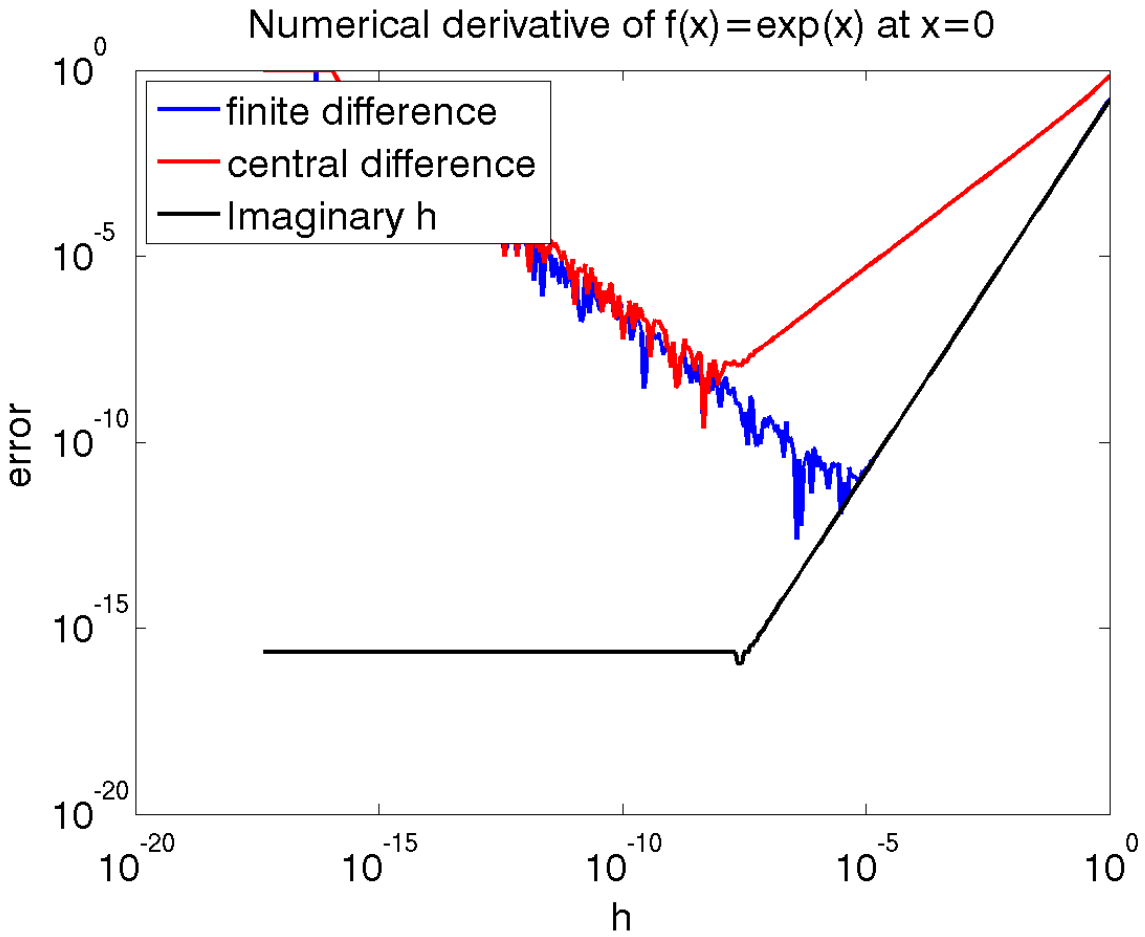
$$\begin{aligned} f'(x+ih) &\approx f(x) + ihf'(x) + \frac{(ih)^2}{2!}f''(x) + \frac{(ih)^3}{3!}f^{(3)}(x) \\ &= f(x) - \frac{h^2}{2}f''(x) + hi(f'(x) - \frac{h^2}{6}f^{(3)}(x)) \end{aligned}$$

Thus the derivative can be approximated by

$$f'(x) \approx \mathrm{Im}\left(\frac{f'(x+ih)}{h}\right)$$

Imaginary Differentiation Error

The graph shows the error in approximating the derivative for $f(x) = e^x$ as a function of h.



By using dual numbers dual numbers $a + bE$, where $E^2 = 0$ we can further reduce the error. For second order derivatives we can use hyper dual numbers. ((2012) J. Fike).

Complex number implementations usually come with most programming languages. For dual or hyper-dual numbers we need to download or implement code.

What is Automatic Differentiation?

Algorithmic Differentiation (often referred to as Automatic Differentiation or just AD) uses the software representation of a function to obtain an efficient method for calculating its derivatives. These derivatives can be of arbitrary order and are analytic in nature (do not have any truncation error). **B. Bell – author of *CppAD***

Use of dual or complex numbers is a form of automatic differentiation. More common though is operator overloading implementations as in `CppAD`.

The Speelpenning function is:

$$f(\mathbf{y}) = \prod_{i=1}^{n} x_i(\mathbf{y})$$

We consider $x_i = (y - t_i)$, so that $f(y) = \prod_{i=1}^{n}(y - t_i)$. If we compute the symbolic derivative we get

$$f'(y) = \sum_{i=1}^{n} \prod_{j \neq i} (y - t_j),$$

which requires $n(n-1)$ multiplications to compute with a straightforward implementation. Rewriting the expression as

$$f'(y) = \sum_{i=1}^{n} \frac{f(y)}{y - t_i},$$

reduces the computational cost, but is not numerically stable if $y \approx t_i$.

Reverse Mode Differentiation

Define $f_k = \prod_{i=1}^{k}(y - t_i)$ and $r_k = \prod_{i=k}^{n}(y - t_i)$, then

$$f'(y) = \sum_{i=1}^{n}\prod_{j\neq i}(y - t_j) = \sum_{i=1}^{n} f_{i-1}r_{i+1},$$

and $f_i = (y - t_i)f_{i-1}$, $f_1 = y - t_1$, $r_i = (y - t_i)r_{i+1}$, $r_n = y - t_n$. Note that $f(y) = f_n = r_1$.

Cost of computing $f(y)$ is n subtractions and $n - 1$ multiplies. At the overhead of storing $f_1, \ldots, f_n$ the derivative can be computed with an additional $n - 1$ additions and $2n - 1$ multiplications.

Reverse Mode Differentiation

Under quite realistic assumptions the evaluation of a gradient requires never more than five times the effort of evaluating the underlying function by itself.

Andreas Griewank (1988)

Reverse mode differentiation (1971) is essentially applying the chain rule on a function in the reverse order that the function is computed.

This is the same technique as applied in the backpropagation algorithm (1969) for neural networks and the forward-backward algorithm for training HMMs (1970).

Reverse mode differentiation applies to very general classes of functions.

Reverse mode differentiation was independently invented by

- G. M. Ostrovskii (1971)
- S. Linnainmaa (1976)
- John Larson and Ahmed Sameh (1978)
- Bert Speelpenning (1980)

Not surprisingly the fame went to B. Speelpenning, who admittedly told the full story at AD 2012.

A. Sameh was on B. Speelpenning's thesis committee.

The Speelpenning function was suggested as a motivation to Speelpenning's thesis by his Canadian interim advisor Prof. Art Sedgwick who passed away on July 5th.

Why worry about matrix differentiation?

The differentiation of functions of a matrix argument is still not well understood

- No simple "calculus" for computing matrix derivatives.
- Matrix derivatives can be complicated to compute even for quite simple expressions.

Goal: Build on known results to define a matrix-derivative calculus.

Quiz

The Tikhonov regularized maximum (log-)likelihood for learning the covariance $\mathbf{\Sigma}$ given an empirical covariance $\mathbf{S}$ is

$$f(\mathbf{\Sigma}) = -\log\det(\mathbf{\Sigma}) - \operatorname{trace}(\mathbf{S}\mathbf{\Sigma}^{-1}) - \|\mathbf{\Sigma}^{-1}\|_F^2.$$

- What is $f'(\mathbf{\Sigma})$ and $f''(\mathbf{\Sigma})$?
- Statisticians can compute it. Can you?
- Can a calculus be reverse engineered?

Terminology

Classical calculus:

- scalar-scalar functions ($f : \mathbb{R} \to \mathbb{R}$)
- scalar-vector functions ($f : \mathbb{R}^d \to \mathbb{R}$).

Matrix calculus:

- scalar-matrix functions ($f : \mathbb{R}^{m \times n} \to \mathbb{R}$)
- matrix-matrix functions ($f : \mathbb{R}^{m \times n} \to \mathbb{R}^{k \times l}$).

Note:

1. Derivatives of scalar-matrix functions requires derivatives of matrix-matrix functions (why?)
2. Matrix-matrix derivatives should be computed implicitly wherever possible (why?)

Optimizing Scalar-Matrix Functions

Useful quantities for optimizing a scalar-matrix function

$$f(\mathbf{X}) : \mathbb{R}^{d \times d} \to \mathbb{R}$$

- The gradient is a matrix–matrix function:

$$f'(\mathbf{X}) : \mathbb{R}^{d \times d} \to \mathbb{R}^{d \times d}.$$

- The Hessian is also a matrix-matrix function:

$$f''(\mathbf{X}) : \mathbb{R}^{d \times d} \to \mathbb{R}^{d^2 \times d^2}.$$

Desiderata: The derivative of a matrix–matrix function should be a matrix, so that we can directly use it to compute the Hessian.

Optimizing Scalar-Matrix Functions (continued)

Taking the scalar–matrix derivative of

$$f(\mathbf{G}(\mathbf{X}))$$

will require the information in the matrix–matrix derivative

$$\frac{\partial \mathbf{G}}{\partial \mathbf{X}}.$$

Desiderata: The derivative of a matrix-matrix function should be a matrix, so that a convenient chain-rule can be established.

The Matrix-Matrix Derivative

We define the matrix–matrix derivative to be:

$$\frac{\partial \mathbf{F}}{\partial \mathbf{X}} \stackrel{\text{def}}{=} \frac{\partial \text{vec}(\mathbf{F}^\top)}{\partial \text{vec}^\top(\mathbf{X}^\top)} = \begin{pmatrix} \frac{\partial f_{11}}{\partial x_{11}} & \frac{\partial f_{11}}{\partial x_{12}} & \cdots & \frac{\partial f_{11}}{\partial x_{mn}} \\ \frac{\partial f_{12}}{\partial x_{11}} & \frac{\partial f_{12}}{\partial x_{12}} & \cdots & \frac{\partial f_{12}}{\partial x_{mn}} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f_{kl}}{\partial x_{11}} & \frac{\partial f_{kl}}{\partial x_{12}} & \cdots & \frac{\partial f_{kl}}{\partial x_{mn}} \end{pmatrix}.$$

The Matrix-Matrix Derivative

We define the matrix–matrix derivative to be:

$$\frac{\partial \mathbf{F}}{\partial \mathbf{X}} \stackrel{\text{def}}{=} \frac{\partial \text{vec}(\mathbf{F}^\top)}{\partial \text{vec}^\top(\mathbf{X}^\top)} = \begin{pmatrix} \frac{\partial f_{11}}{\partial x_{11}} & \frac{\partial f_{11}}{\partial x_{12}} & \cdots & \frac{\partial f_{11}}{\partial x_{mn}} \\ \frac{\partial f_{12}}{\partial x_{11}} & \frac{\partial f_{12}}{\partial x_{12}} & \cdots & \frac{\partial f_{12}}{\partial x_{mn}} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f_{kl}}{\partial x_{11}} & \frac{\partial f_{kl}}{\partial x_{12}} & \cdots & \frac{\partial f_{kl}}{\partial x_{mn}} \end{pmatrix}.$$

The Matrix-Matrix Derivative

We define the matrix–matrix derivative to be:

$$\frac{\partial \mathbf{F}}{\partial \mathbf{X}} \stackrel{\text{def}}{=} \frac{\partial \text{vec}(\mathbf{F}^\top)}{\partial \text{vec}^\top(\mathbf{X}^\top)} = \begin{pmatrix} \frac{\partial f_{11}}{\partial x_{11}} & \frac{\partial f_{11}}{\partial x_{12}} & \cdots & \frac{\partial f_{11}}{\partial x_{mn}} \\ \frac{\partial f_{12}}{\partial x_{11}} & \frac{\partial f_{12}}{\partial x_{12}} & \cdots & \frac{\partial f_{12}}{\partial x_{mn}} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f_{kl}}{\partial x_{11}} & \frac{\partial f_{kl}}{\partial x_{12}} & \cdots & \frac{\partial f_{kl}}{\partial x_{mn}} \end{pmatrix}.$$

The Matrix-Matrix Derivative

We define the matrix–matrix derivative to be:

$$\frac{\partial \mathbf{F}}{\partial \mathbf{X}} \stackrel{\text{def}}{=} \frac{\partial \text{vec}(\mathbf{F}^\top)}{\partial \text{vec}^\top(\mathbf{X}^\top)} = \begin{pmatrix} \frac{\partial f_{11}}{\partial x_{11}} & \frac{\partial f_{11}}{\partial x_{12}} & \cdots & \frac{\partial f_{11}}{\partial x_{mn}} \\ \frac{\partial f_{12}}{\partial x_{11}} & \frac{\partial f_{12}}{\partial x_{12}} & \cdots & \frac{\partial f_{12}}{\partial x_{mn}} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f_{kl}}{\partial x_{11}} & \frac{\partial f_{kl}}{\partial x_{12}} & \cdots & \frac{\partial f_{kl}}{\partial x_{mn}} \end{pmatrix}.$$

Note that the matrix–matrix derivative of a scalar–matrix function is not the same as the scalar–matrix derivative:

$$\frac{\partial \text{ mat}(f)}{\partial \mathbf{X}} = \text{vec}^\top \left(\left(\frac{\partial f}{\partial \mathbf{X}} \right)^\top \right).$$

Derivatives of Linear Matrix Functions

1. A matrix–matrix derivative is a matrix outer-product:

$$F(\mathbf{X}) = \mathrm{trace}(\mathbf{AX})\mathbf{B}, \qquad \frac{\partial F}{\partial \mathbf{X}} = \mathrm{vec}(\mathbf{B}^\top)\mathrm{vec}^\top(\mathbf{A}).$$

2. A matrix–matrix derivative is a Kronecker product:

$$F(\mathbf{X}) = \mathbf{AXB}, \qquad \frac{\partial F}{\partial \mathbf{X}} = \mathbf{A} \otimes \mathbf{B}^\top.$$

3. A matrix–matrix derivative *cannot* be expressed with standard operations. We call the new operation the box product

$$F(\mathbf{X}) = \mathbf{AX}^\top\mathbf{B}, \qquad \frac{\partial F}{\partial \mathbf{X}} = \mathbf{A} \boxtimes \mathbf{B}^\top.$$

Direct Matrix Products

A Direct Matrix Product, $\mathbf{X} = \mathbf{A} \circledast \mathbf{B}$, is a matrix whose elements are

$$x_{(i_1 i_2)(i_3 i_4)} = a_{i_{\sigma(1)} i_{\sigma(2)}} b_{i_{\sigma(3)} i_{\sigma(4)}}.$$

- $(i_1 i_2)$ is shorthand for $i_1 n_2 + i_2 - 1$ where $i_2 \in \{1, 2, \ldots, n_2\}$.
- σ is a permutation over $\mathbb{Z}_4$

The direct matrix products are central to matrix–matrix differentiation. 8 can be expressed using Kronecker products:

$$\begin{array}{cccc} \mathbf{A} \otimes \mathbf{B} & \mathbf{A}^\top \otimes \mathbf{B} & \mathbf{A} \otimes \mathbf{B}^\top & \mathbf{A}^\top \otimes \mathbf{B}^\top \\ \mathbf{B} \otimes \mathbf{A} & \mathbf{B}^\top \otimes \mathbf{A} & \mathbf{B} \otimes \mathbf{A}^\top & \mathbf{B}^\top \otimes \mathbf{A}^\top \end{array}$$

Then 8 using matrix outer products and the final 8 using the box product.

Definition

Let $\mathbf{A} \in \mathbb{R}^{m_1 \times n_1}$ and $\mathbf{B} \in \mathbb{R}^{m_2 \times n_2}$

Definition (Kronecker Product)

$\mathbf{A} \otimes \mathbf{B} \in \mathbb{R}^{(m_1 m_2) \times (n_1 n_2)}$ is defined by $(\mathbf{A} \otimes \mathbf{B})_{(i-1)m_2+j,(k-1)n_2+l} = a_{ik} b_{jl} = (\mathbf{A} \otimes \mathbf{B})_{(ij)(kl)}$.

Definition (Box Product)

$\mathbf{A} \boxtimes \mathbf{B} \in \mathbb{R}^{(m_1 m_2) \times (n_1 n_2)}$ is defined by $(\mathbf{A} \boxtimes \mathbf{B})_{(i-1)m_2+j,(k-1)n_1+l} = a_{il} b_{jk} = (\mathbf{A} \boxtimes \mathbf{B})_{(ij)(kl)}$.

An example Kronecker and box product

To see the structure consider the box product of two 2×2 matrices, $\mathbf{A}$ and $\mathbf{B}$:

$$\mathbf{A} \otimes \mathbf{B}$$

$$\begin{pmatrix} a_{11}b_{11} & a_{11}b_{12} & a_{12}b_{11} & a_{12}b_{12} \\ a_{11}b_{21} & a_{11}b_{22} & a_{12}b_{21} & a_{12}b_{22} \\ a_{21}b_{11} & a_{21}b_{12} & a_{22}b_{11} & a_{22}b_{12} \\ a_{21}b_{21} & a_{21}b_{22} & a_{22}b_{21} & a_{22}b_{22} \end{pmatrix}$$

$$\mathbf{A} \boxtimes \mathbf{B}$$

$$\begin{pmatrix} a_{11}b_{11} & a_{12}b_{11} & a_{11}b_{12} & a_{12}b_{12} \\ a_{11}b_{21} & a_{12}b_{21} & a_{11}b_{22} & a_{12}b_{22} \\ a_{21}b_{11} & a_{22}b_{11} & a_{21}b_{12} & a_{22}b_{12} \\ a_{21}b_{21} & a_{22}b_{21} & a_{21}b_{22} & a_{22}b_{22} \end{pmatrix}$$

An example Kronecker and box product

To see the structure consider the box product of two 2×2 matrices, $\mathbf{A}$ and $\mathbf{B}$:

$$\mathbf{A} \otimes \mathbf{B}$$

$$\begin{pmatrix} a_{11}b_{11} & a_{11}b_{12} & a_{12}b_{11} & a_{12}b_{12} \\ a_{11}b_{21} & a_{11}b_{22} & a_{12}b_{21} & a_{12}b_{22} \\ a_{21}b_{11} & a_{21}b_{12} & a_{22}b_{11} & a_{22}b_{12} \\ a_{21}b_{21} & a_{21}b_{22} & a_{22}b_{21} & a_{22}b_{22} \end{pmatrix}$$

$$\mathbf{A} \boxtimes \mathbf{B}$$

$$\begin{pmatrix} a_{11}b_{11} & a_{12}b_{11} & a_{11}b_{12} & a_{12}b_{12} \\ a_{11}b_{21} & a_{12}b_{21} & a_{11}b_{22} & a_{12}b_{22} \\ a_{21}b_{11} & a_{22}b_{11} & a_{21}b_{12} & a_{22}b_{12} \\ a_{21}b_{21} & a_{22}b_{21} & a_{21}b_{22} & a_{22}b_{22} \end{pmatrix}$$

An example Kronecker and box product

To see the structure consider the box product of two 2×2 matrices, $\mathbf{A}$ and $\mathbf{B}$:

$$\mathbf{A} \otimes \mathbf{B}$$

$$\begin{pmatrix} a_{11}b_{11} & a_{11}b_{12} & a_{12}b_{11} & a_{12}b_{12} \\ a_{11}b_{21} & a_{11}b_{22} & a_{12}b_{21} & a_{12}b_{22} \\ a_{21}b_{11} & a_{21}b_{12} & a_{22}b_{11} & a_{22}b_{12} \\ a_{21}b_{21} & a_{21}b_{22} & a_{22}b_{21} & a_{22}b_{22} \end{pmatrix}$$

$$\mathbf{A} \boxtimes \mathbf{B}$$

$$\begin{pmatrix} a_{11}b_{11} & a_{12}b_{11} & a_{11}b_{12} & a_{12}b_{12} \\ a_{11}b_{21} & a_{12}b_{21} & a_{11}b_{22} & a_{12}b_{22} \\ a_{21}b_{11} & a_{22}b_{11} & a_{21}b_{12} & a_{22}b_{12} \\ a_{21}b_{21} & a_{22}b_{21} & a_{21}b_{22} & a_{22}b_{22} \end{pmatrix}$$

An example Kronecker and box product

To see the structure consider the box product of two 2×2 matrices, $\mathbf{A}$ and $\mathbf{B}$:

$$\mathbf{A} \otimes \mathbf{B}$$

$$\begin{pmatrix} a_{11}b_{11} & a_{11}b_{12} & a_{12}b_{11} & a_{12}b_{12} \\ a_{11}b_{21} & a_{11}b_{22} & a_{12}b_{21} & a_{12}b_{22} \\ a_{21}b_{11} & a_{21}b_{12} & a_{22}b_{11} & a_{22}b_{12} \\ a_{21}b_{21} & a_{21}b_{22} & a_{22}b_{21} & a_{22}b_{22} \end{pmatrix}$$

$$\mathbf{A} \boxtimes \mathbf{B}$$

$$\begin{pmatrix} a_{11}b_{11} & a_{12}b_{11} & a_{11}b_{12} & a_{12}b_{12} \\ a_{11}b_{21} & a_{12}b_{21} & a_{11}b_{22} & a_{12}b_{22} \\ a_{21}b_{11} & a_{22}b_{11} & a_{21}b_{12} & a_{22}b_{12} \\ a_{21}b_{21} & a_{22}b_{21} & a_{21}b_{22} & a_{22}b_{22} \end{pmatrix}$$

An example Kronecker and box product

To see the structure consider the box product of two 2×2 matrices, $\mathbf{A}$ and $\mathbf{B}$:

$$\mathbf{A} \otimes \mathbf{B}$$

$$\begin{pmatrix} a_{11}b_{11} & a_{11}b_{12} & a_{12}b_{11} & a_{12}b_{12} \\ a_{11}b_{21} & a_{11}b_{22} & a_{12}b_{21} & a_{12}b_{22} \\ a_{21}b_{11} & a_{21}b_{12} & a_{22}b_{11} & a_{22}b_{12} \\ a_{21}b_{21} & a_{21}b_{22} & a_{22}b_{21} & a_{22}b_{22} \end{pmatrix}$$

$$\mathbf{A} \boxtimes \mathbf{B}$$

$$\begin{pmatrix} a_{11}b_{11} & a_{12}b_{11} & a_{11}b_{12} & a_{12}b_{12} \\ a_{11}b_{21} & a_{12}b_{21} & a_{11}b_{22} & a_{12}b_{22} \\ a_{21}b_{11} & a_{22}b_{11} & a_{21}b_{12} & a_{22}b_{12} \\ a_{21}b_{21} & a_{22}b_{21} & a_{21}b_{22} & a_{22}b_{22} \end{pmatrix}$$

An example Kronecker and box product

To see the structure consider the box product of two 2×2 matrices, $\mathbf{A}$ and $\mathbf{B}$:

$$\mathbf{A} \otimes \mathbf{B}$$

$$\begin{pmatrix} a_{11}b_{11} & a_{11}b_{12} & a_{12}b_{11} & a_{12}b_{12} \\ a_{11}b_{21} & a_{11}b_{22} & a_{12}b_{21} & a_{12}b_{22} \\ a_{21}b_{11} & a_{21}b_{12} & a_{22}b_{11} & a_{22}b_{12} \\ a_{21}b_{21} & a_{21}b_{22} & a_{22}b_{21} & a_{22}b_{22} \end{pmatrix}$$

$$\mathbf{A} \boxtimes \mathbf{B}$$

$$\begin{pmatrix} a_{11}b_{11} & a_{12}b_{11} & a_{11}b_{12} & a_{12}b_{12} \\ a_{11}b_{21} & a_{12}b_{21} & a_{11}b_{22} & a_{12}b_{22} \\ a_{21}b_{11} & a_{22}b_{11} & a_{21}b_{12} & a_{22}b_{12} \\ a_{21}b_{21} & a_{22}b_{21} & a_{21}b_{22} & a_{22}b_{22} \end{pmatrix}$$

An example Kronecker and box product

To see the structure consider the box product of two 2×2 matrices, $\mathbf{A}$ and $\mathbf{B}$:

$$\mathbf{A} \otimes \mathbf{B}$$

$$\begin{pmatrix} a_{11}b_{11} & a_{11}b_{12} & a_{12}b_{11} & a_{12}b_{12} \\ a_{11}b_{21} & a_{11}b_{22} & a_{12}b_{21} & a_{12}b_{22} \\ a_{21}b_{11} & a_{21}b_{12} & a_{22}b_{11} & a_{22}b_{12} \\ a_{21}b_{21} & a_{21}b_{22} & a_{22}b_{21} & a_{22}b_{22} \end{pmatrix}$$

$$\mathbf{A} \boxtimes \mathbf{B}$$

$$\begin{pmatrix} a_{11}b_{11} & a_{12}b_{11} & a_{11}b_{12} & a_{12}b_{12} \\ a_{11}b_{21} & a_{12}b_{21} & a_{11}b_{22} & a_{12}b_{22} \\ a_{21}b_{11} & a_{22}b_{11} & a_{21}b_{12} & a_{22}b_{12} \\ a_{21}b_{21} & a_{22}b_{21} & a_{21}b_{22} & a_{22}b_{22} \end{pmatrix}$$

An example Kronecker and box product

To see the structure consider the box product of two 2×2 matrices, $\mathbf{A}$ and $\mathbf{B}$:

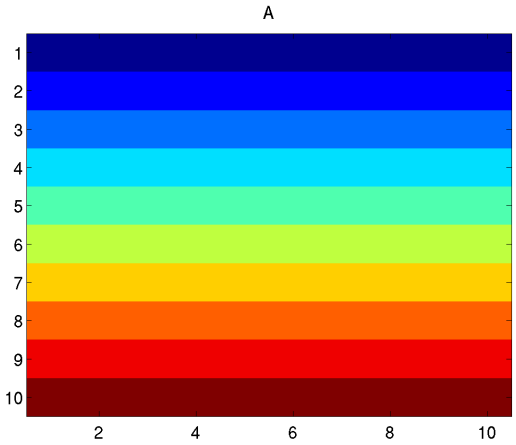
$$\mathbf{A} \otimes \mathbf{B}$$

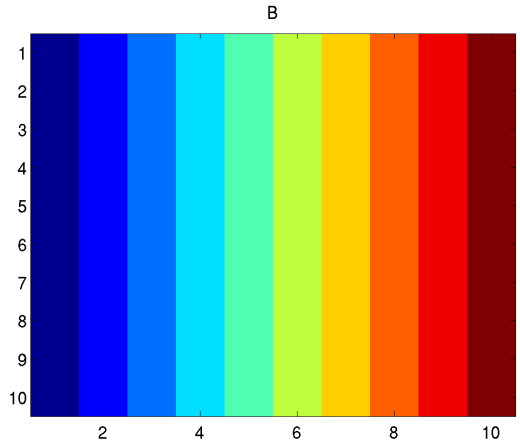
$$\begin{pmatrix} a_{11}b_{11} & a_{11}b_{12} & a_{12}b_{11} & a_{12}b_{12} \\ a_{11}b_{21} & a_{11}b_{22} & a_{12}b_{21} & a_{12}b_{22} \\ a_{21}b_{11} & a_{21}b_{12} & a_{22}b_{11} & a_{22}b_{12} \\ a_{21}b_{21} & a_{21}b_{22} & a_{22}b_{21} & a_{22}b_{22} \end{pmatrix}$$

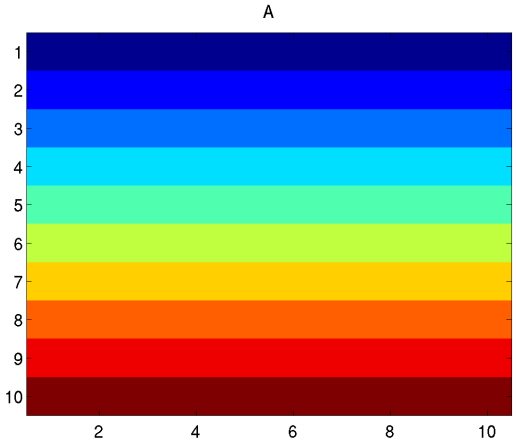
$$\mathbf{A} \boxtimes \mathbf{B}$$

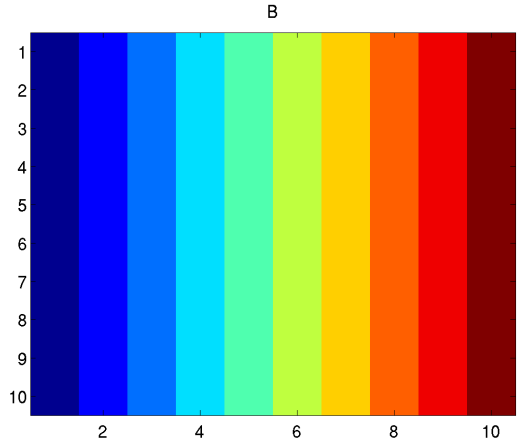
$$\begin{pmatrix} a_{11}b_{11} & a_{12}b_{11} & a_{11}b_{12} & a_{12}b_{12} \\ a_{11}b_{21} & a_{12}b_{21} & a_{11}b_{22} & a_{12}b_{22} \\ a_{21}b_{11} & a_{22}b_{11} & a_{21}b_{12} & a_{22}b_{12} \\ a_{21}b_{21} & a_{22}b_{21} & a_{21}b_{22} & a_{22}b_{22} \end{pmatrix}$$

$$\mathbf{A} = \begin{pmatrix} 1 & 1 & 1 & \cdots & 1 \\ 2 & 2 & 2 & \cdots & 2 \\ 3 & 3 & 3 & \cdots & 3 \\ \vdots & \vdots & \vdots & & \vdots \\ 10 & 10 & 10 & \cdots & 10 \end{pmatrix} \qquad \mathbf{B} = \begin{pmatrix} 1 & 2 & 3 & \cdots & 10 \\ 1 & 2 & 3 & \cdots & 10 \\ 1 & 2 & 3 & \cdots & 10 \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & 2 & 3 & \cdots & 10 \end{pmatrix}$$

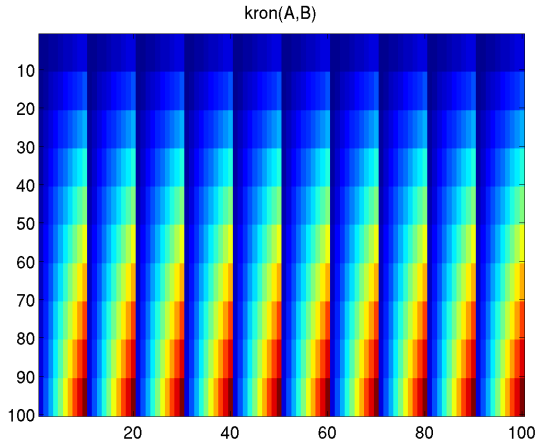

A
1
2
3
4
5
6
7
8
9
10
2
4
6
8
10

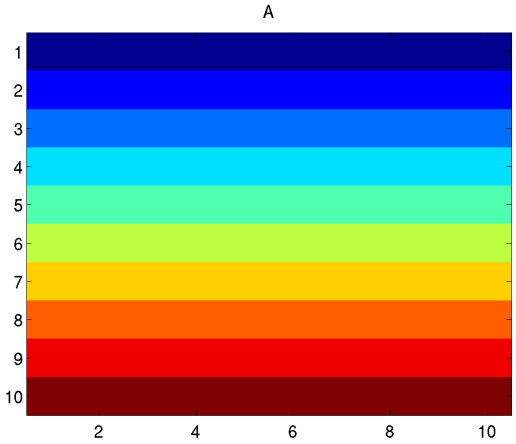

B
1
2
3
4
5
6
7
8
9
10
2
4
6
8
10

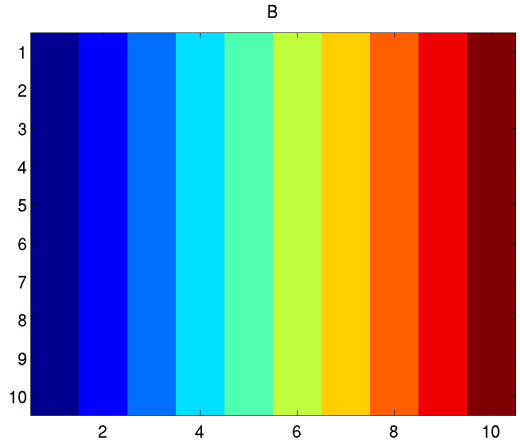




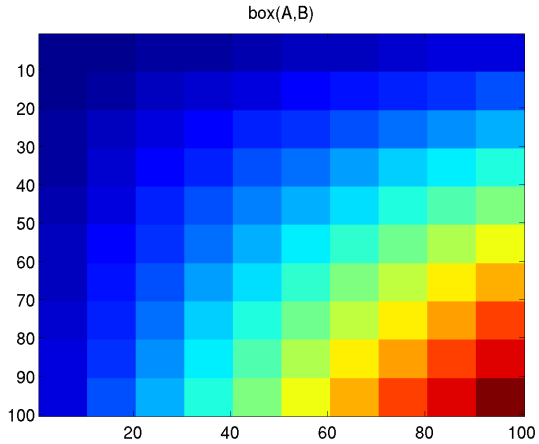
$$\mathbf{A} \otimes \mathbf{B}$$

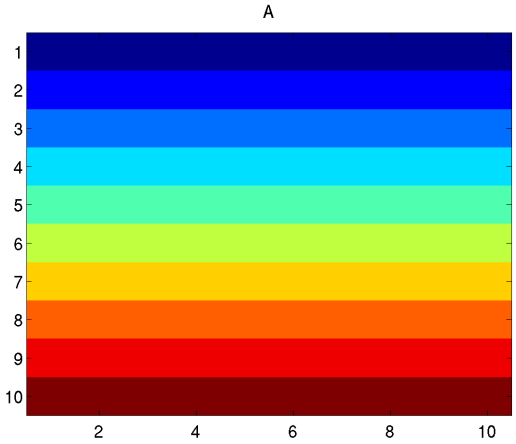


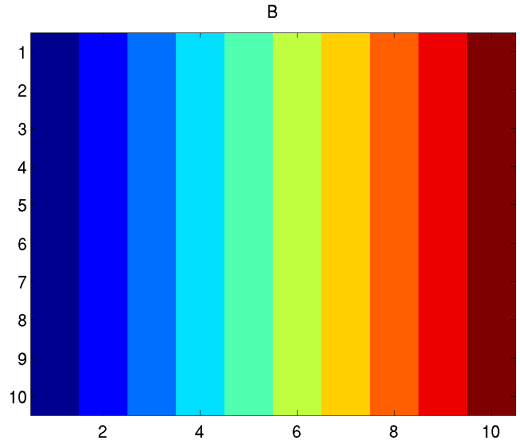




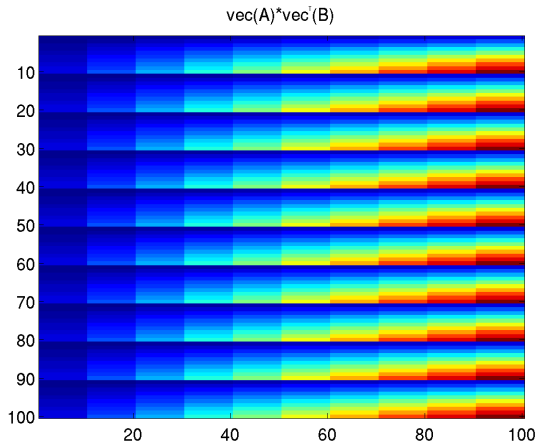
$$\mathbf{A} \boxtimes \mathbf{B}$$







$$\text{vec}(\mathbf{A})\text{vec}^{\top}(\mathbf{B})$$



The box product behaves similarly to the Kronecker product:

1. **Vector Multiplication**:

$$(\mathbf{B}^\top \otimes \mathbf{A})\mathrm{vec}(\mathbf{X}) = \mathrm{vec}(\mathbf{AXB}) \quad (\mathbf{B}^\top \boxtimes \mathbf{A})\mathrm{vec}(\mathbf{X}) = \mathrm{vec}(\mathbf{AX}^\top\mathbf{B})$$

The box product behaves similarly to the Kronecker product:

1. **Vector Multiplication**:

$$(\mathbf{B}^\top \otimes \mathbf{A})\mathrm{vec}(\mathbf{X}) = \mathrm{vec}(\mathbf{AXB}) \quad (\mathbf{B}^\top \boxtimes \mathbf{A})\mathrm{vec}(\mathbf{X}) = \mathrm{vec}(\mathbf{AX}^\top\mathbf{B})$$

2. **Matrix Multiplication**:

$$(\mathbf{A} \otimes \mathbf{B})(\mathbf{C} \otimes \mathbf{D}) = (\mathbf{AC}) \otimes (\mathbf{BD}) \quad (\mathbf{A} \boxtimes \mathbf{B})(\mathbf{C} \boxtimes \mathbf{D}) = (\mathbf{AD}) \otimes (\mathbf{BC})$$

The box product behaves similarly to the Kronecker product:

1. **Vector Multiplication**:

$$(\mathbf{B}^\top \otimes \mathbf{A})\mathrm{vec}(\mathbf{X}) = \mathrm{vec}(\mathbf{AXB}) \quad (\mathbf{B}^\top \boxtimes \mathbf{A})\mathrm{vec}(\mathbf{X}) = \mathrm{vec}(\mathbf{AX}^\top\mathbf{B})$$

2. **Matrix Multiplication**:

$$(\mathbf{A} \otimes \mathbf{B})(\mathbf{C} \otimes \mathbf{D}) = (\mathbf{AC}) \otimes (\mathbf{BD}) \quad (\mathbf{A} \boxtimes \mathbf{B})(\mathbf{C} \boxtimes \mathbf{D}) = (\mathbf{AD}) \otimes (\mathbf{BC})$$

3. **Inverse and Transpose**:

$$(\mathbf{A} \otimes \mathbf{B})^{-1} = \mathbf{A}^{-1} \otimes \mathbf{B}^{-1} \quad (\mathbf{A} \boxtimes \mathbf{B})^{-1} = \mathbf{B}^{-1} \boxtimes \mathbf{A}^{-1}$$
$$(\mathbf{A} \otimes \mathbf{B})^\top = \mathbf{A}^\top \otimes \mathbf{B}^\top \quad (\mathbf{A} \boxtimes \mathbf{B})^\top = \mathbf{B}^\top \boxtimes \mathbf{A}^\top$$

The box product behaves similarly to the Kronecker product:

1. **Vector Multiplication**:

$$(\mathbf{B}^\top \otimes \mathbf{A})\mathrm{vec}(\mathbf{X}) = \mathrm{vec}(\mathbf{AXB}) \quad (\mathbf{B}^\top \boxtimes \mathbf{A})\mathrm{vec}(\mathbf{X}) = \mathrm{vec}(\mathbf{AX}^\top\mathbf{B})$$

2. **Matrix Multiplication**:

$$(\mathbf{A} \otimes \mathbf{B})(\mathbf{C} \otimes \mathbf{D}) = (\mathbf{AC}) \otimes (\mathbf{BD}) \quad (\mathbf{A} \boxtimes \mathbf{B})(\mathbf{C} \boxtimes \mathbf{D}) = (\mathbf{AD}) \otimes (\mathbf{BC})$$

3. **Inverse and Transpose**:

$$(\mathbf{A} \otimes \mathbf{B})^{-1} = \mathbf{A}^{-1} \otimes \mathbf{B}^{-1} \quad (\mathbf{A} \boxtimes \mathbf{B})^{-1} = \mathbf{B}^{-1} \boxtimes \mathbf{A}^{-1}$$
$$(\mathbf{A} \otimes \mathbf{B})^\top = \mathbf{A}^\top \otimes \mathbf{B}^\top \quad (\mathbf{A} \boxtimes \mathbf{B})^\top = \mathbf{B}^\top \boxtimes \mathbf{A}^\top$$

4. **Mixed Products**:

$$(\mathbf{A} \otimes \mathbf{B})(\mathbf{C} \boxtimes \mathbf{D}) = (\mathbf{AC}) \boxtimes (\mathbf{BD}) \quad (\mathbf{A} \boxtimes \mathbf{B})(\mathbf{C} \otimes \mathbf{D}) = (\mathbf{AD}) \boxtimes (\mathbf{BC})$$

Let $\mathbf{A} \in \mathbb{R}^{m_1 \times n_1}$, $\mathbf{B} \in \mathbb{R}^{m_2 \times n_2}$ then

- **Trace:**

$$\operatorname{trace}(\mathbf{A} \otimes \mathbf{B}) = \operatorname{trace}(\mathbf{A}) \operatorname{trace}(\mathbf{B}) \qquad \operatorname{trace}(\mathbf{A} \boxtimes \mathbf{B}) = \operatorname{trace}(\mathbf{AB})$$

- **Determinant:** Here $m_1 = n_1$ and $m_2 = n_2$ is required

$$\begin{aligned} \det(\mathbf{A} \otimes \mathbf{B}) &= (\det(\mathbf{A}))^{m_2} (\det(\mathbf{A}))^{m_1} \\ \det(\mathbf{A} \boxtimes \mathbf{B}) &= (-1)^{\binom{m_1}{2}\binom{m_2}{2}} (\det(\mathbf{A}))^{m_2} (\det(\mathbf{B}))^{m_1} \end{aligned}$$

- **Associativity:**

$$(\mathbf{A} \otimes \mathbf{B}) \otimes \mathbf{C} = \mathbf{A} \otimes (\mathbf{B} \otimes \mathbf{C}) \qquad (\mathbf{A} \boxtimes \mathbf{B}) \boxtimes \mathbf{C} = \mathbf{A} \boxtimes (\mathbf{B} \boxtimes \mathbf{C}),$$

but not for mixed products. In general we have

$$(\mathbf{A} \otimes \mathbf{B}) \boxtimes \mathbf{C} \neq \mathbf{A} \otimes (\mathbf{B} \boxtimes \mathbf{C}) \qquad (\mathbf{A} \boxtimes \mathbf{B}) \otimes \mathbf{C} \neq \mathbf{A} \boxtimes (\mathbf{B} \otimes \mathbf{C}).$$

Identity Box Products

The identity box product $\mathbf{I}_m \boxtimes \mathbf{I}_n$ is permutation matrix with interesting properties. Let $\mathbf{A} \in \mathbb{R}^{m_1 \times n_1}$, $\mathbf{B} \in \mathbb{R}^{m_2 \times n_2}$.

Orthonormal: $(\mathbf{I}_m \boxtimes \mathbf{I}_n)^\top (\mathbf{I}_m \boxtimes \mathbf{I}_n) = \mathbf{I}_{mn}$

Transposition: $(\mathbf{I}_{m_1} \boxtimes \mathbf{I}_{n_1})\mathrm{vec}(\mathbf{A}) = \mathrm{vec}(\mathbf{A}^\top)$

Connector: Converting a Kronecker product to a box product: $(\mathbf{A} \otimes \mathbf{B})(\mathbf{I}_{n_1} \boxtimes \mathbf{I}_{n_2}) = \mathbf{A} \boxtimes \mathbf{B}$.
Converting a box product to a Kronecker product: $(\mathbf{A} \boxtimes \mathbf{B})(\mathbf{I}_{n_2} \boxtimes \mathbf{I}_{n_1}) = \mathbf{A} \otimes \mathbf{B}$.

Box products are old things in a new wrapping

Although the notation for box products is new, $\mathbf{I}_m \boxtimes \mathbf{I}_n$ has long been known as $\mathbf{T}_{m,n}$ by physicists, or as a stride permutation $\mathbf{L}_m^{mn}$ by others. These objects are identical, but the box-product allows us to express more complex identities more compactly, e.g. let $\mathbf{A} \in \mathbb{R}^{m_1 \times n_1}$, $\mathbf{B} \in \mathbb{R}^{m_2 \times n_2}$ then

$$(\mathbf{I}_{m_1} \boxtimes \mathbf{I}_{n_1 m_2} \boxtimes \mathbf{I}_{n_2})\mathrm{vec}((\mathbf{A} \otimes \mathbf{B})^\top) = \mathrm{vec}(\mathrm{vec}(\mathbf{A})\mathrm{vec}^\top(\mathbf{B})).$$

The initial permutation matrix would have to be written

$$\mathbf{I}_{m_1} \boxtimes \mathbf{I}_{n_1 m_2} \boxtimes \mathbf{I}_{n_2} = (\mathbf{I}_{m_1} \otimes \mathbf{T}_{n_1 m_2, m_1})\mathbf{T}_{m_1, n_1 m_1 m_2},$$

if the notation of box-products wasn't being used.

Direct matrix products are important because such matrices can be multiplied fast with each other and with vectors.

Let us show how the FFT can be done in terms of Kronecker and box products. Recall that the DFT matrix of order n is given by

$\mathbf{F}_n = [e^{-2\pi ikl/n}]_{0\leq k,l<n} = [\omega_n^{kl}]_{0\leq k,l<n}$ and therefore

$$\mathbf{F}_2 = \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}, \qquad \mathbf{F}_4 = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -i & -1 & i \\ 1 & -1 & 1 & -1 \\ 1 & i & -1 & -i \end{pmatrix}$$

We can factor the matrix $\mathbf{F}_4$ as follows:

$$\mathbf{F}_4 = \begin{pmatrix} 1 & & 1 & \\ & 1 & & 1 \\ 1 & & -1 & \\ & 1 & & -1 \end{pmatrix} \begin{pmatrix} 1 & & & \\ & 1 & & \\ & & 1 & \\ & & & -i \end{pmatrix} \begin{pmatrix} 1 & 1 & & \\ 1 & -1 & & \\ & & 1 & 1 \\ & & 1 & -1 \end{pmatrix} \begin{pmatrix} 1 & & & \\ & & 1 & \\ & 1 & & \\ & & & 1 \end{pmatrix}$$

or more compactly

$$\mathbf{F}_4 = (\mathbf{F}_2 \otimes \mathbf{I}_2)\mathrm{diag}\left(\mathrm{vec}\begin{pmatrix} 1 & 1 \\ 1 & -i \end{pmatrix}\right)(\mathbf{I}_2 \boxtimes \mathbf{F}_2)$$

In general if we define the matrix

$$\mathbf{V}_{N,M}(\alpha) = \begin{pmatrix} 1 & 1 & 1 & \dots & 1 \\ 1 & \alpha & \alpha^2 & \dots & \alpha^{M-1} \\ 1 & \alpha^2 & \alpha^4 & \dots & \alpha^{2(M-1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \alpha^{N-1} & \alpha^{2(N-1)} & \dots & \alpha^{(N-1)(M-1)} \end{pmatrix},$$

For $N = km$ we have the following factorizations of the DFT matrix:

$$\begin{aligned} \mathbf{F}_N &= (\mathbf{F}_k \otimes \mathbf{I}_m)(\mathrm{diag}(\mathrm{vec}(\mathbf{V}_{m,k}(\omega_N))))(\mathbf{I}_k \boxtimes \mathbf{F}_m) \\ \mathbf{F}_N &= (\mathbf{F}_m \boxtimes \mathbf{I}_k)(\mathrm{diag}(\mathrm{vec}(\mathbf{V}_{m,k}(\omega_N))))(\mathbf{F}_k \otimes \mathbf{I}_m) \end{aligned}$$

This allows $\mathrm{FFT}_N(\mathbf{x}) = \mathbf{y} = \mathbf{F}_N\mathbf{x}$ to be computed as

$$\mathbf{F}_N\mathbf{x} = \mathrm{vec}((\mathbf{V}_{m,k}(\omega_N) \circ (\mathbf{F}_m\mathbf{X}^\top))\mathbf{F}_k^\top).$$

where $\mathbf{x} = \mathrm{vec}(\mathbf{X})$ and $\circ$ denotes elementwise multiplication ($.*$ in Matlab). The direct computation has a cost of $\mathcal{O}(k^2m^2)$, whereas the formula above does the job in $\mathcal{O}(km(k+m))$ operations. Repeated use of the identity for $N = 2^n$ leads to the Cooley-Tukey FFT algorithm. (James Cooley was at T. J. Watson from 1962 to 1991).

The fastest FFT library in the world as of 2012 (Spiral) uses knowledge of several such factorizations to automatically optimize the FFT implementation for an arbitrary value of n to a given platform!

Here we show the role of Kronecker and box product in the matrix–matrix differentiation framework.

Basic Differentiation Identitities

Let $\mathbf{X} \in \mathbb{R}^{m \times n}$.

- **Identity**: $\mathbf{F}(\mathbf{X}) = \mathbf{X}$, $\mathbf{F}'(\mathbf{X}) = \mathbf{I}_{mn}$
- **Transpose**:$\mathbf{F}(\mathbf{X}) = \mathbf{X}^\top$, $\mathbf{F}'(\mathbf{X}) = \mathbf{I}_m \boxtimes \mathbf{I}_n$
- **Chain Rule**:

$$\mathbf{F}(\mathbf{X}) = \mathbf{G}(\mathbf{H}(\mathbf{X})), \quad \mathbf{F}'(\mathbf{X}) = \mathbf{G}'(\mathbf{H}(\mathbf{X}))\mathbf{H}'(\mathbf{X})$$

- **Product Rule**:

$$\mathbf{F}(\mathbf{X}) = \mathbf{G}(\mathbf{X})\mathbf{H}(\mathbf{X}), \quad \mathbf{F}'(\mathbf{X}) = (\mathbf{I} \otimes \mathbf{H}^\top(\mathbf{X}))\mathbf{G}'(\mathbf{X}) + (\mathbf{G}(\mathbf{X}) \otimes \mathbf{I})\mathbf{H}'(\mathbf{X})$$

More Derivative Identitities

Assume $\mathbf{X} \in \mathbb{R}^{m \times m}$ is a square matrix.

- **Square:** $\mathbf{F}(\mathbf{X}) = \mathbf{X}^2$, $\mathbf{F}'(\mathbf{X}) = \mathbf{I}_m \otimes \mathbf{X}^\top + \mathbf{X} \otimes \mathbf{I}_m$.
- **Inverse:** $\mathbf{F}(\mathbf{X}) = \mathbf{X}^{-1}$, $\mathbf{F}'(\mathbf{X}) = -\mathbf{X}^{-1} \otimes \mathbf{X}^{-\top}$.
- **+Transpose:** $\mathbf{F}(\mathbf{X}) = \mathbf{X}^{-\top}$, $\mathbf{F}'(\mathbf{X}) = -\mathbf{X}^{-\top} \boxtimes \mathbf{X}^{-1}$.
- **Square Root:** $\mathbf{F}(\mathbf{X}) = \mathbf{X}^{1/2}$,
  $\mathbf{F}'(\mathbf{X}) = \left(\mathbf{I} \otimes (\mathbf{X}^{1/2})^\top + \mathbf{X}^{1/2} \otimes \mathbf{I}\right)^{-1}$.
- **Integer Power:** $\mathbf{F}(\mathbf{X}) = \mathbf{X}^k$, $\mathbf{F}'(\mathbf{X}) = \sum_{i=0}^{k-1} \mathbf{X}^i \otimes (\mathbf{X}^{k-1-i})^\top$.

First some simple identities:

$$\begin{aligned} f(\mathbf{X}) &= \operatorname{trace}(\mathbf{A}\mathbf{X}) & f'(\mathbf{X}) &= \mathbf{A}^\top \\ f(\mathbf{X}) &= \operatorname{trace}(\mathbf{A}\mathbf{X}^\top) & f'(\mathbf{X}) &= \mathbf{A} \\ f(\mathbf{X}) &= \log\det(\mathbf{X}) & f'(\mathbf{X}) &= \mathbf{X}^{-\top} \end{aligned}$$

Then the chain rule for more general expressions:

$$\begin{aligned} \operatorname{vec}\left(\left(\frac{\partial}{\partial \mathbf{X}}\log\det(\mathbf{G}(\mathbf{X}))\right)^\top\right) &= \left(\frac{\partial \mathbf{G}}{\partial \mathbf{X}}\right)^\top \operatorname{vec}\left((\mathbf{G}(\mathbf{X}))^{-1}\right) \\ \operatorname{vec}\left(\left(\frac{\partial}{\partial \mathbf{X}}\operatorname{trace}(\mathbf{G}(\mathbf{X}))\right)^\top\right) &= \left(\frac{\partial \mathbf{G}}{\partial \mathbf{X}}\right)^\top \operatorname{vec}(\mathbf{I}) \end{aligned}$$

The chain rule to get scalar-matrix derivatives is awkward to use. Instead we have some short-cuts.

$$\frac{\partial}{\partial \mathbf{X}} \operatorname{trace}(\mathbf{G}(\mathbf{X})\mathbf{H}(\mathbf{X})) = \frac{\partial}{\partial \mathbf{X}} \operatorname{trace}\left(\mathbf{H}(\mathbf{Y})\mathbf{G}(\mathbf{X}) + \mathbf{H}(\mathbf{X})\mathbf{G}(\mathbf{Y})\right)\Big|_{\mathbf{Y}=\mathbf{X}},$$

$$\frac{\partial}{\partial \mathbf{X}} \operatorname{trace}(\mathbf{A}\mathbf{F}^{-1}(\mathbf{X})) = -\frac{\partial}{\partial \mathbf{X}} \operatorname{trace}\left(\mathbf{F}^{-1}(\mathbf{Y})\mathbf{A}\mathbf{F}^{-1}(\mathbf{Y})\mathbf{F}(\mathbf{X})\right)\Big|_{\mathbf{Y}=\mathbf{X}},$$

and

$$\frac{\partial \log\det(\mathbf{F}(\mathbf{X}))}{\partial \mathbf{X}} = \frac{\partial}{\partial \mathbf{X}} \operatorname{trace}\left(\mathbf{F}^{-1}(\mathbf{Y})\mathbf{F}(\mathbf{X})\right)\Big|_{\mathbf{Y}=\mathbf{X}}.$$

Use the formulas on the previous page to compute

$$\frac{\partial}{\partial \mathbf{X}} \operatorname{trace}((\mathbf{X}-\mathbf{A})(\mathbf{X}-\mathbf{B})^{-1}(\mathbf{X}-\mathbf{C}))$$

Use the formulas on the previous page to compute

$$\frac{\partial}{\partial \mathbf{X}} \operatorname{trace}((\mathbf{X}-\mathbf{A})(\mathbf{X}-\mathbf{B})^{-1}(\mathbf{X}-\mathbf{C}))$$

The answer is:

$$(\mathbf{X}-\mathbf{C})^{\top}(\mathbf{X}-\mathbf{B})^{-\top}+(\mathbf{X}-\mathbf{B})^{-\top}(\mathbf{X}-\mathbf{A})^{\top}$$
$$+(\mathbf{X}-\mathbf{B})^{-\top}(\mathbf{X}-\mathbf{A})^{\top}(\mathbf{X}-\mathbf{C})^{\top}(\mathbf{X}-\mathbf{B})^{-\top}$$

Finally if

$$r(x) = \frac{q(x)}{p(x)} = \frac{\sum_{i=1}^{n} a_i x^i}{\sum_{j=1}^{n} b_i x^i}$$

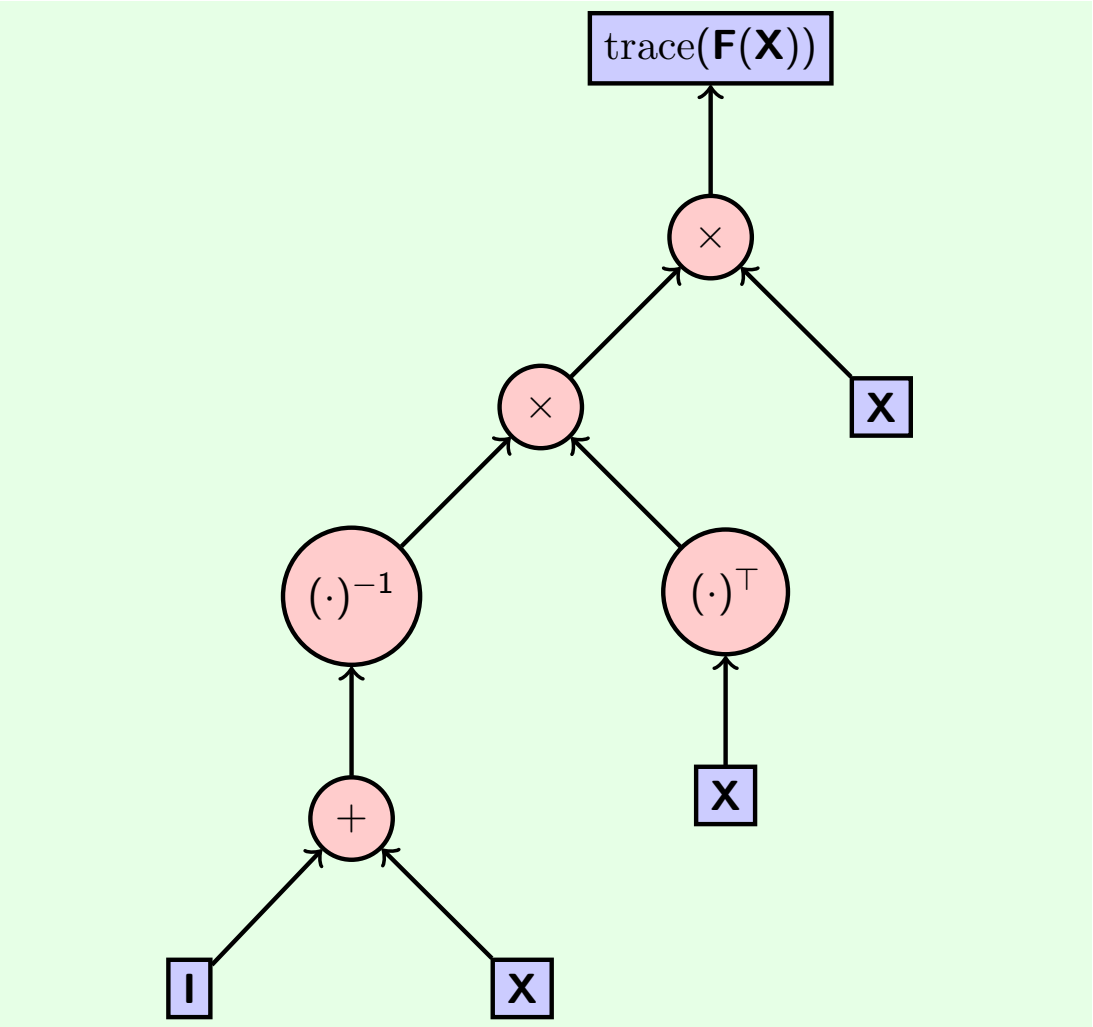
is a scalar-scalar function we can form a matrix-matrix function by simply substituting a matrix $\mathbf{X}$ for the scalar x. Then

$$\frac{\partial}{\partial \mathbf{X}} \operatorname{trace}(r(\mathbf{X})) = \left(r'(\mathbf{X})\right)^\top$$

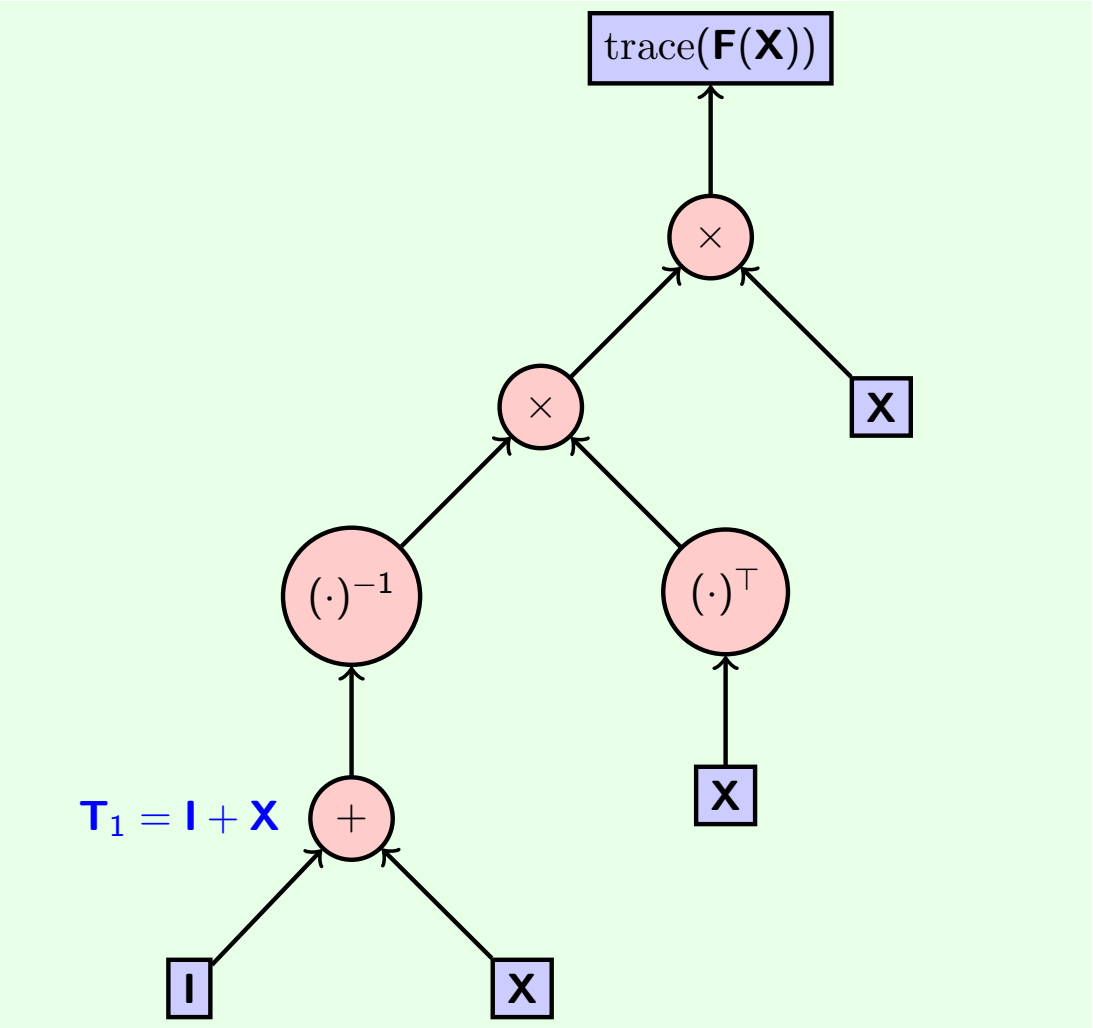
and if $h(x) = \log(r(x))$ then

$$\frac{\partial}{\partial \mathbf{X}} \log \det(r(\mathbf{X})) = \left(h'(\mathbf{X})\right)^\top.$$

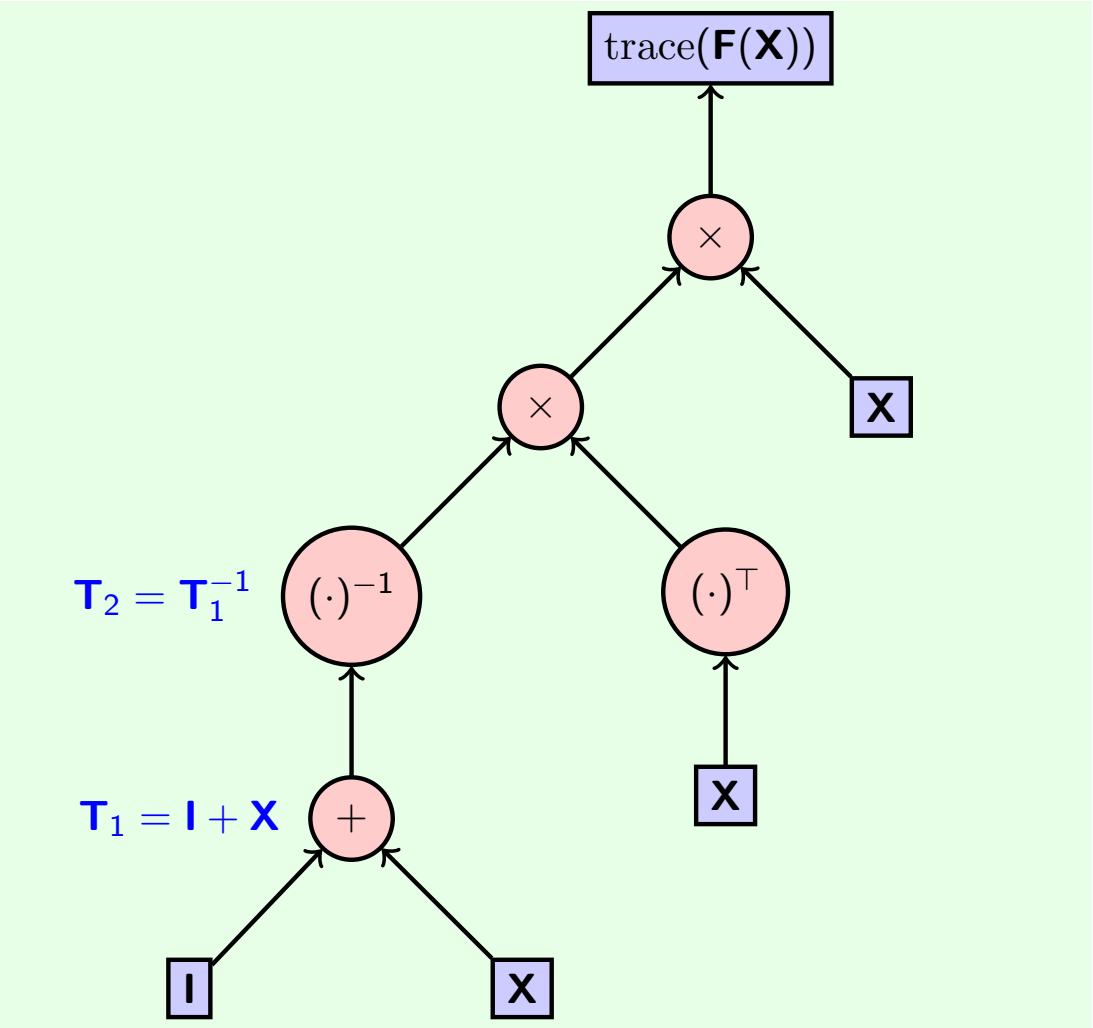
Compute $f(\mathbf{X}) = \mathrm{trace}(\mathbf{F}(\mathbf{X})) = \mathrm{trace}(((\mathbf{I} + \mathbf{X})^{-1}\mathbf{X}^\top)\mathbf{X})$



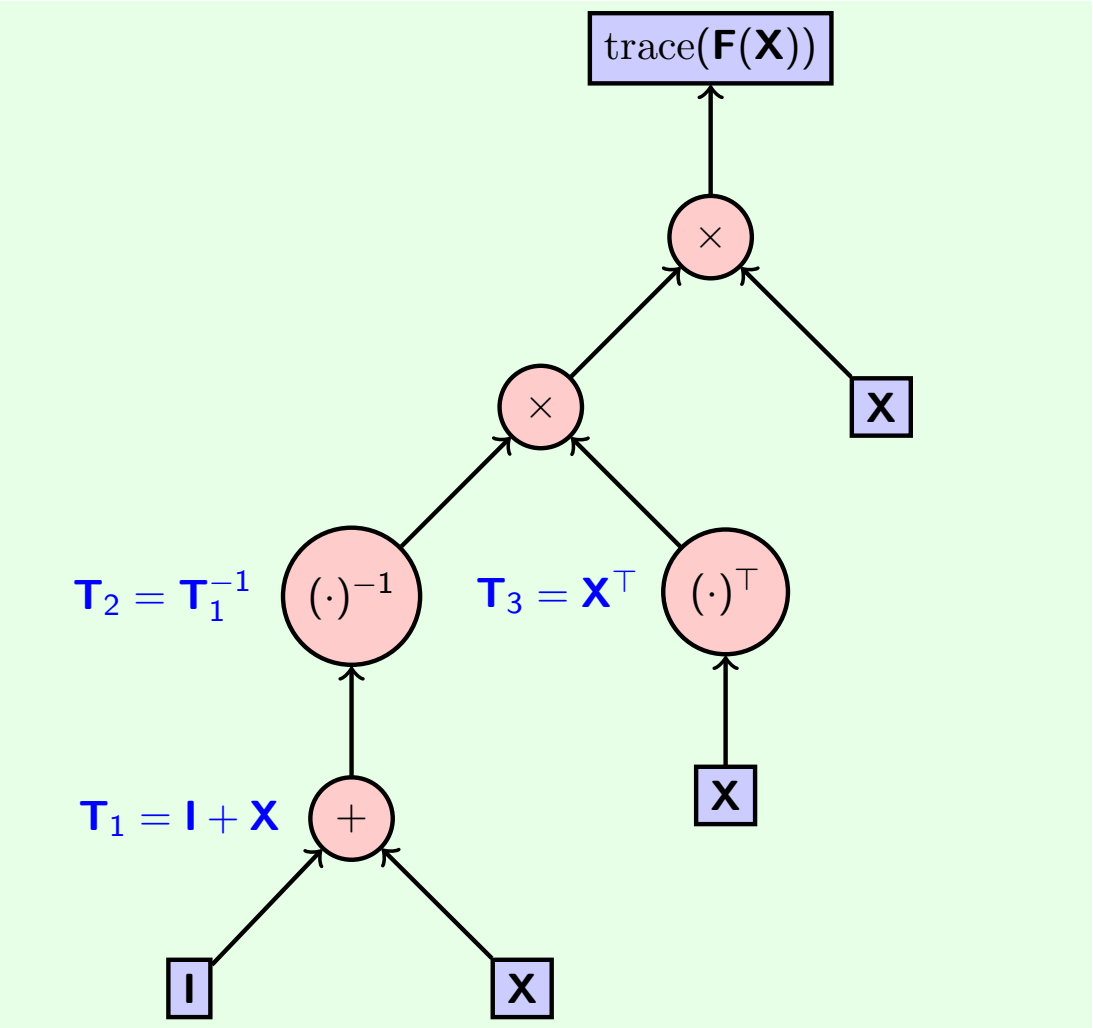
Compute $f(\mathbf{X}) = \mathrm{trace}(\mathbf{F}(\mathbf{X})) = \mathrm{trace}(((\mathbf{I}+\mathbf{X})^{-1}\mathbf{X}^\top)\mathbf{X})$



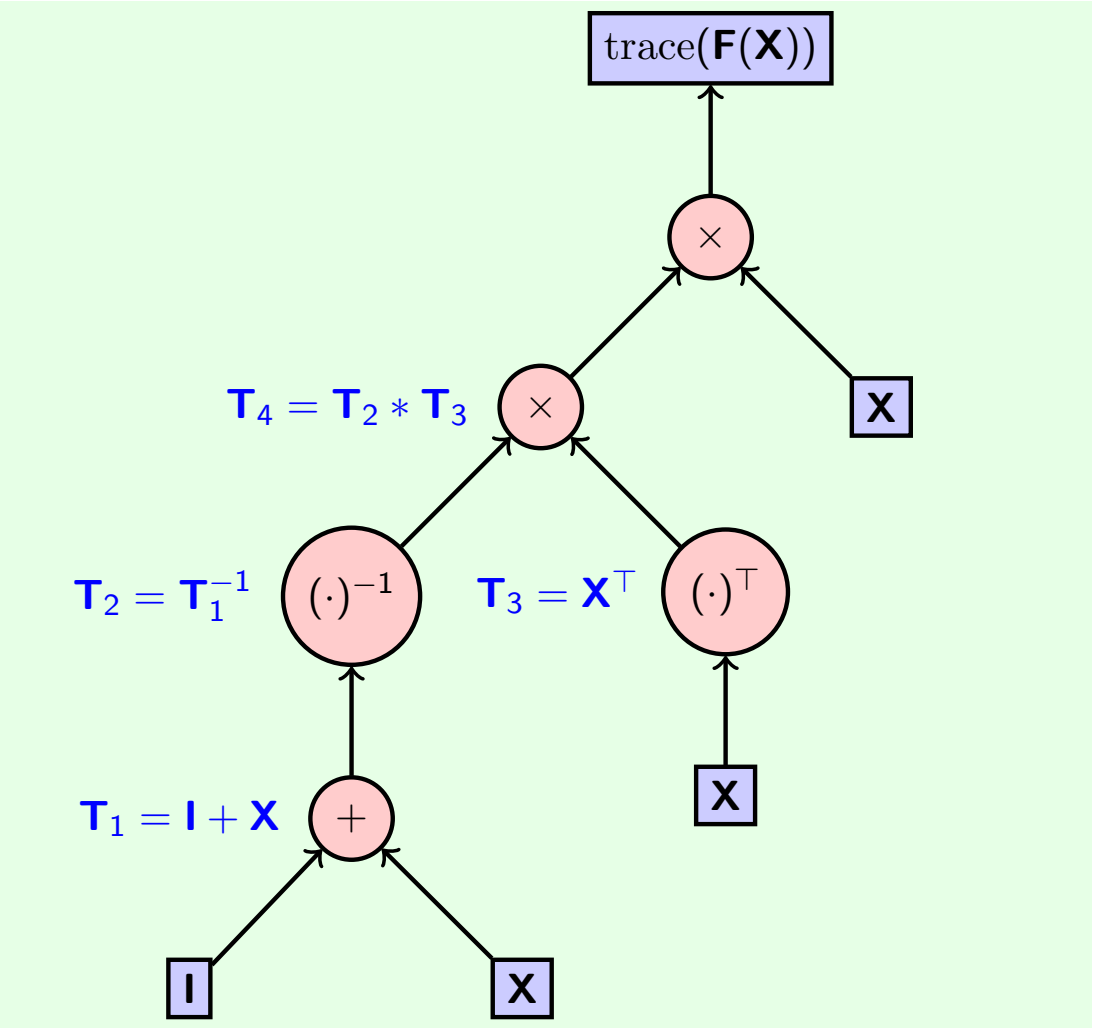
Compute $f(\mathbf{X}) = \mathrm{trace}(\mathbf{F}(\mathbf{X})) = \mathrm{trace}(((\mathbf{I}+\mathbf{X})^{-1}\mathbf{X}^\top)\mathbf{X})$



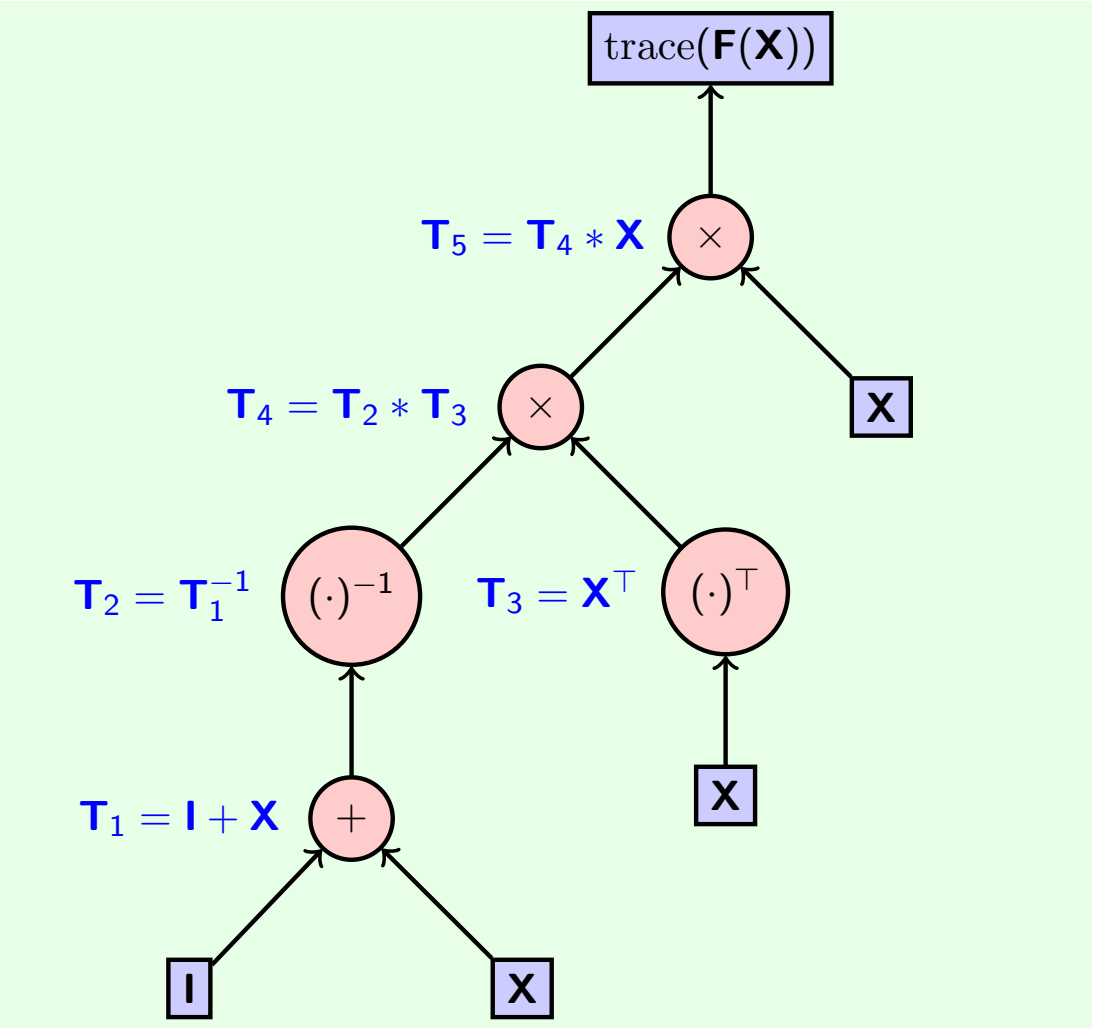
Compute $f(\mathbf{X}) = \mathrm{trace}(\mathbf{F}(\mathbf{X})) = \mathrm{trace}(((\mathbf{I}+\mathbf{X})^{-1}\mathbf{X}^\top)\mathbf{X})$



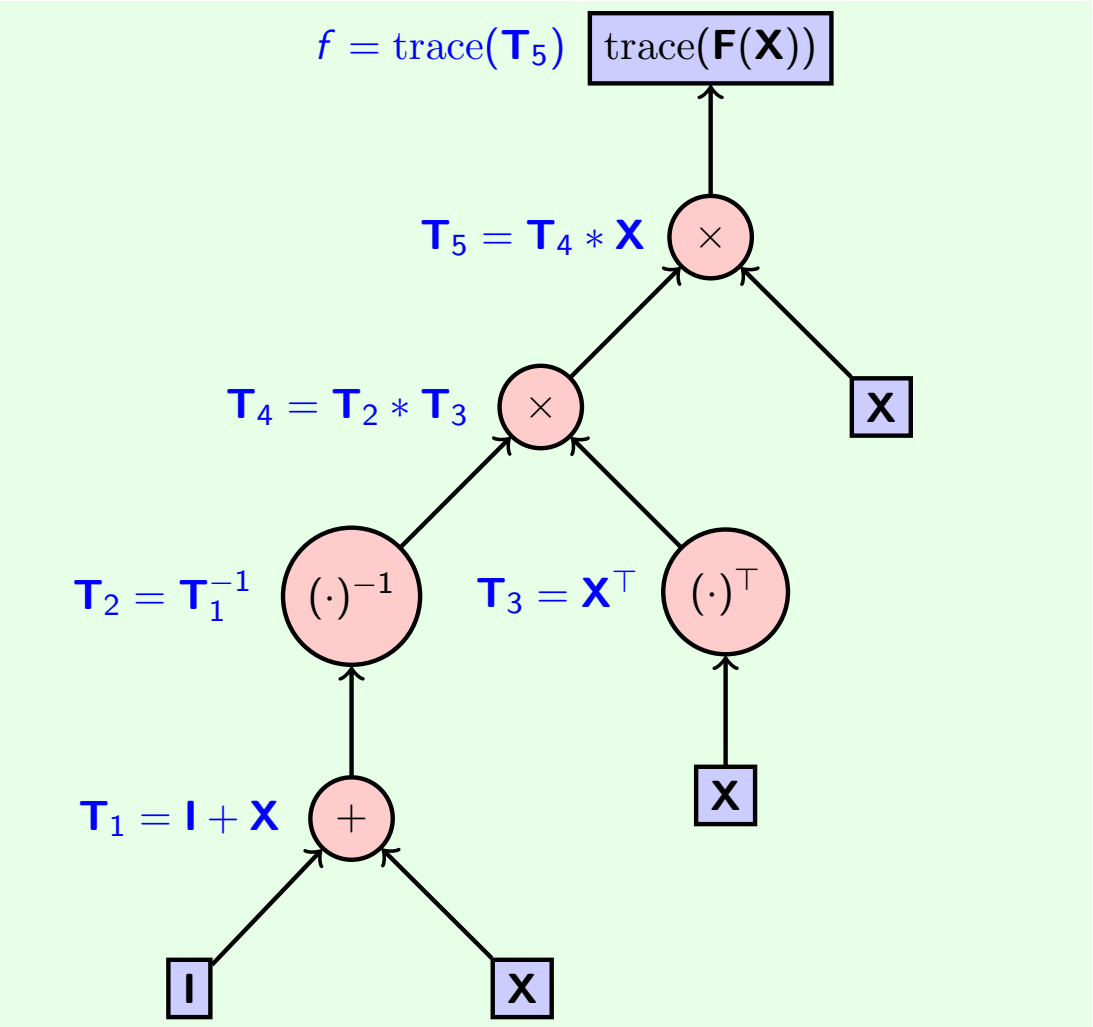
Compute $f(\mathbf{X}) = \mathrm{trace}(\mathbf{F}(\mathbf{X})) = \mathrm{trace}(((\mathbf{I} + \mathbf{X})^{-1}\mathbf{X}^\top)\mathbf{X})$



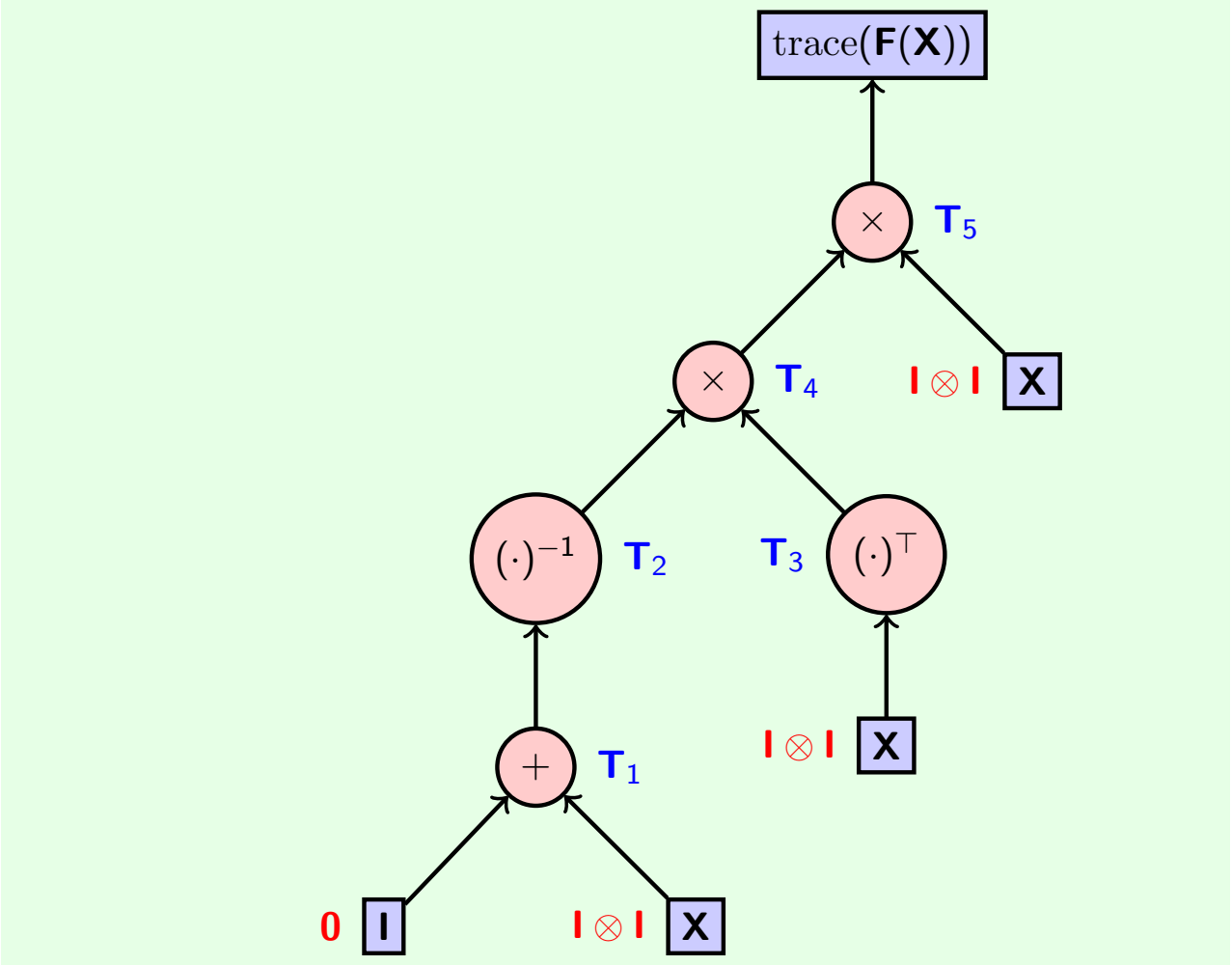
Compute $f(\mathbf{X}) = \mathrm{trace}(\mathbf{F}(\mathbf{X})) = \mathrm{trace}(((\mathbf{I} + \mathbf{X})^{-1}\mathbf{X}^\top)\mathbf{X})$



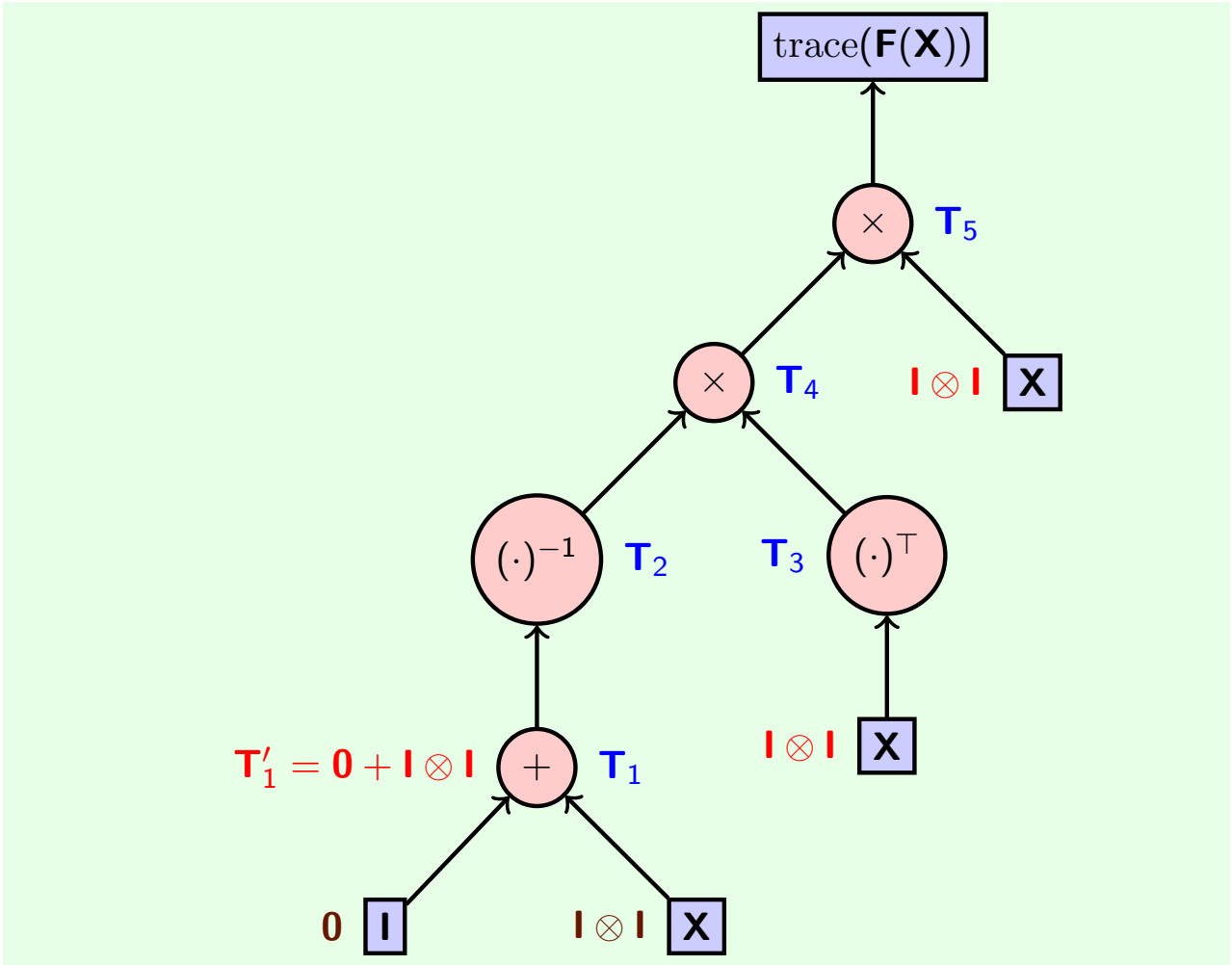
Compute $f(\mathbf{X}) = \text{trace}(\mathbf{F}(\mathbf{X})) = \text{trace}(((\mathbf{I} + \mathbf{X})^{-1}\mathbf{X}^\top)\mathbf{X})$



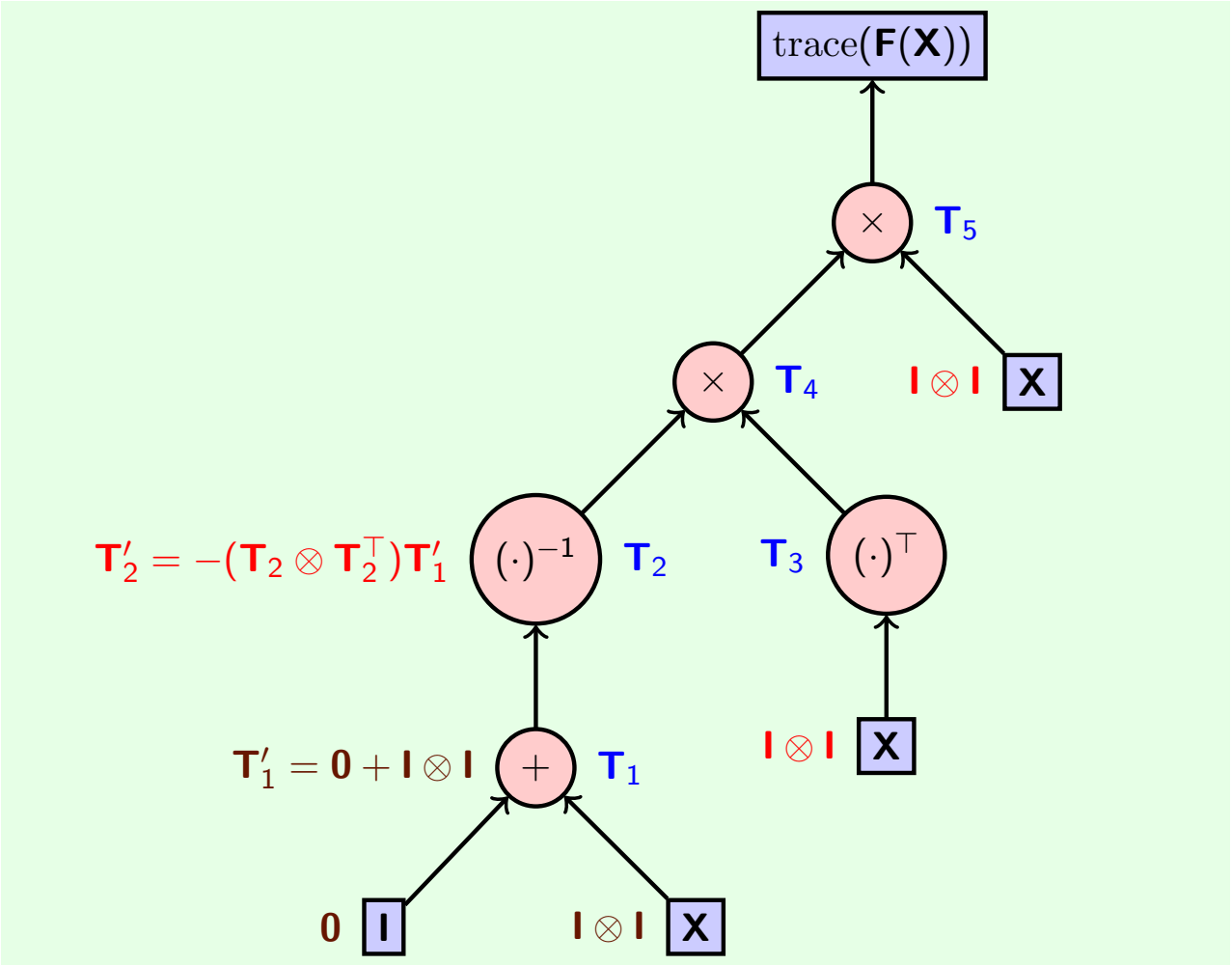
Forward mode computation of $f'(\mathbf{X})$:



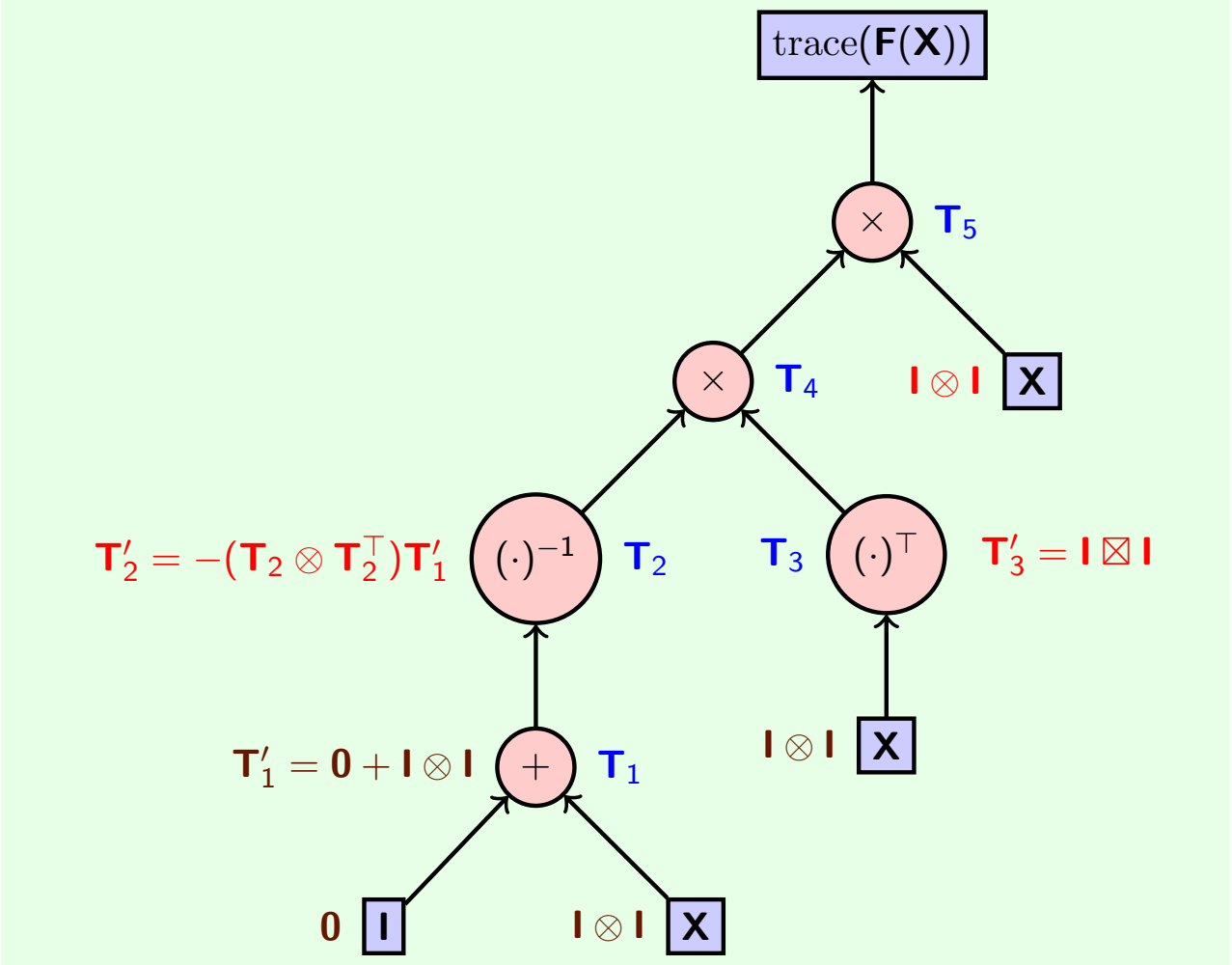
$$(\mathbf{G} + \mathbf{H})' = \mathbf{G}' + \mathbf{H}'$$



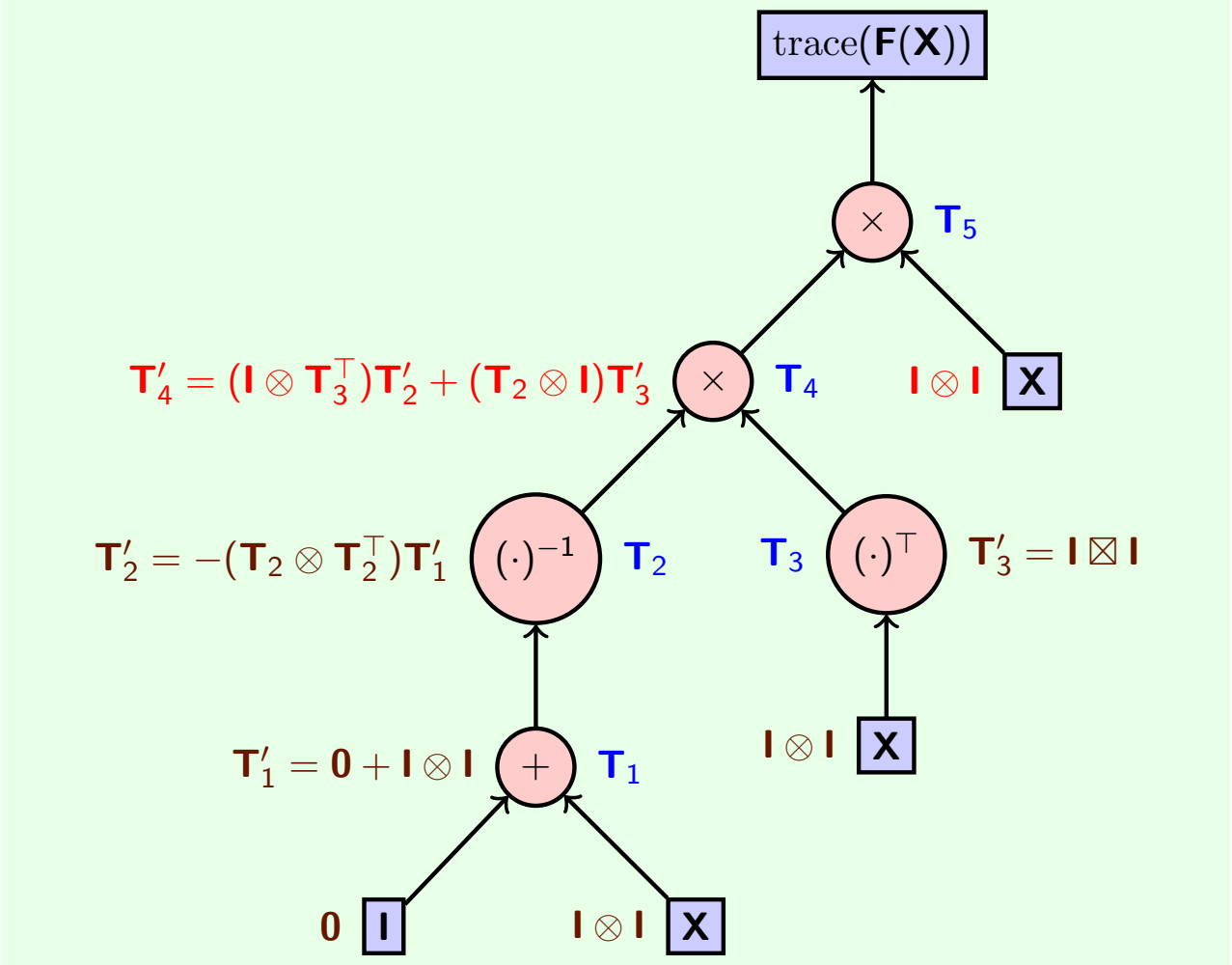
Chain rule: $(\mathbf{T}_1^{-1})' = -(\mathbf{T}_1^{-1} \otimes \mathbf{T}_1^{-\top})\mathbf{T}_1'$



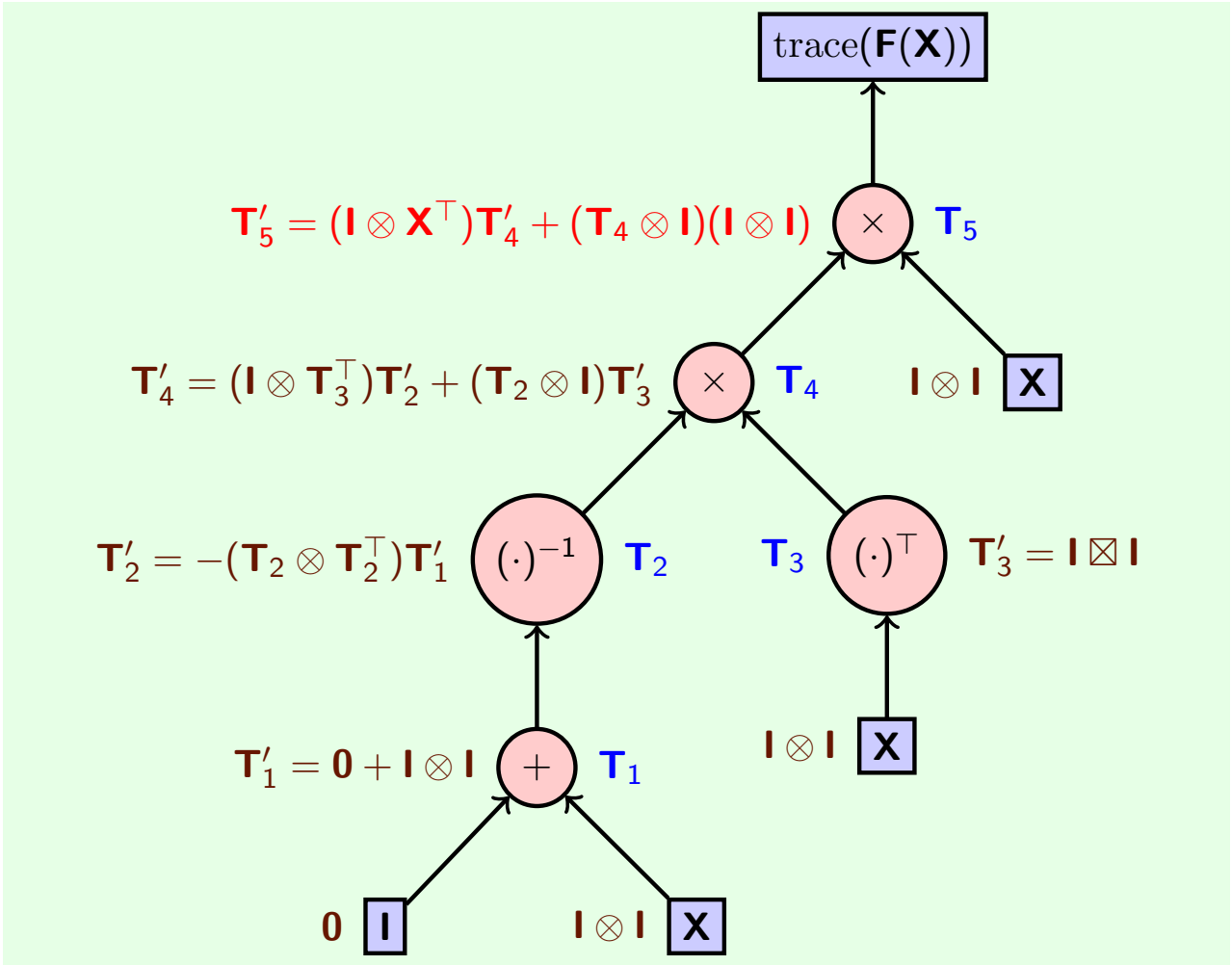
$$(\mathbf{X}^\top)' = \mathbf{I} \boxtimes \mathbf{I}$$



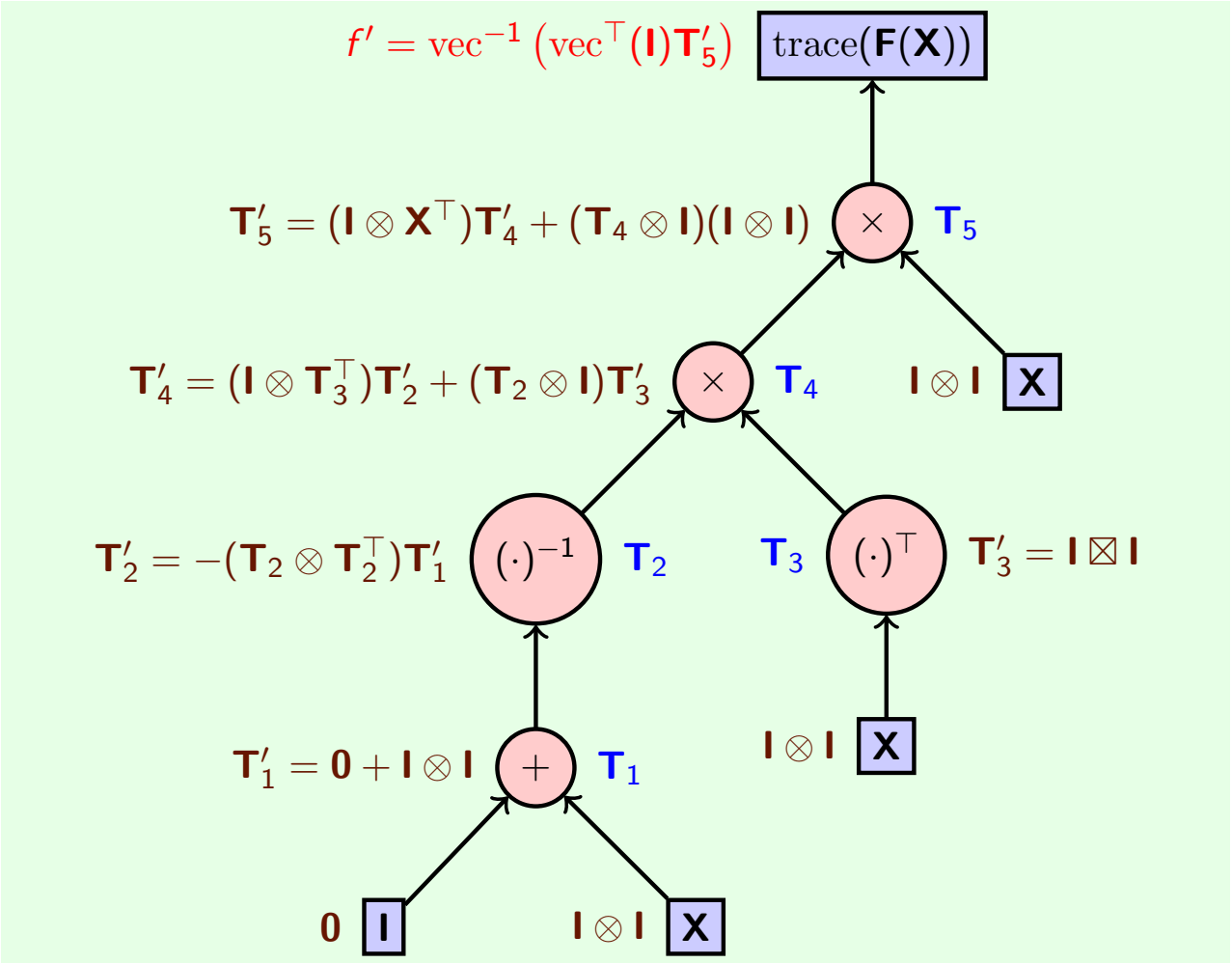
Product rule: $(\mathbf{GH})' = (\mathbf{I} \otimes \mathbf{H}^\top)\mathbf{G}' + (\mathbf{G} \otimes \mathbf{I})\mathbf{H}'$



Product rule: $(\mathbf{GH})' = (\mathbf{I} \otimes \mathbf{H}^\top)\mathbf{G}' + (\mathbf{G} \otimes \mathbf{I})\mathbf{H}'$



$$((\text{trace}(\mathbf{F}))')^\top = \text{vec}^{-1}\left(\text{vec}^\top(\mathbf{I})\mathbf{F}'\right)$$



Forward mode differentiation

- Requires huge intermediate matrices.
- Only the last step reduces the size.

Critical points:

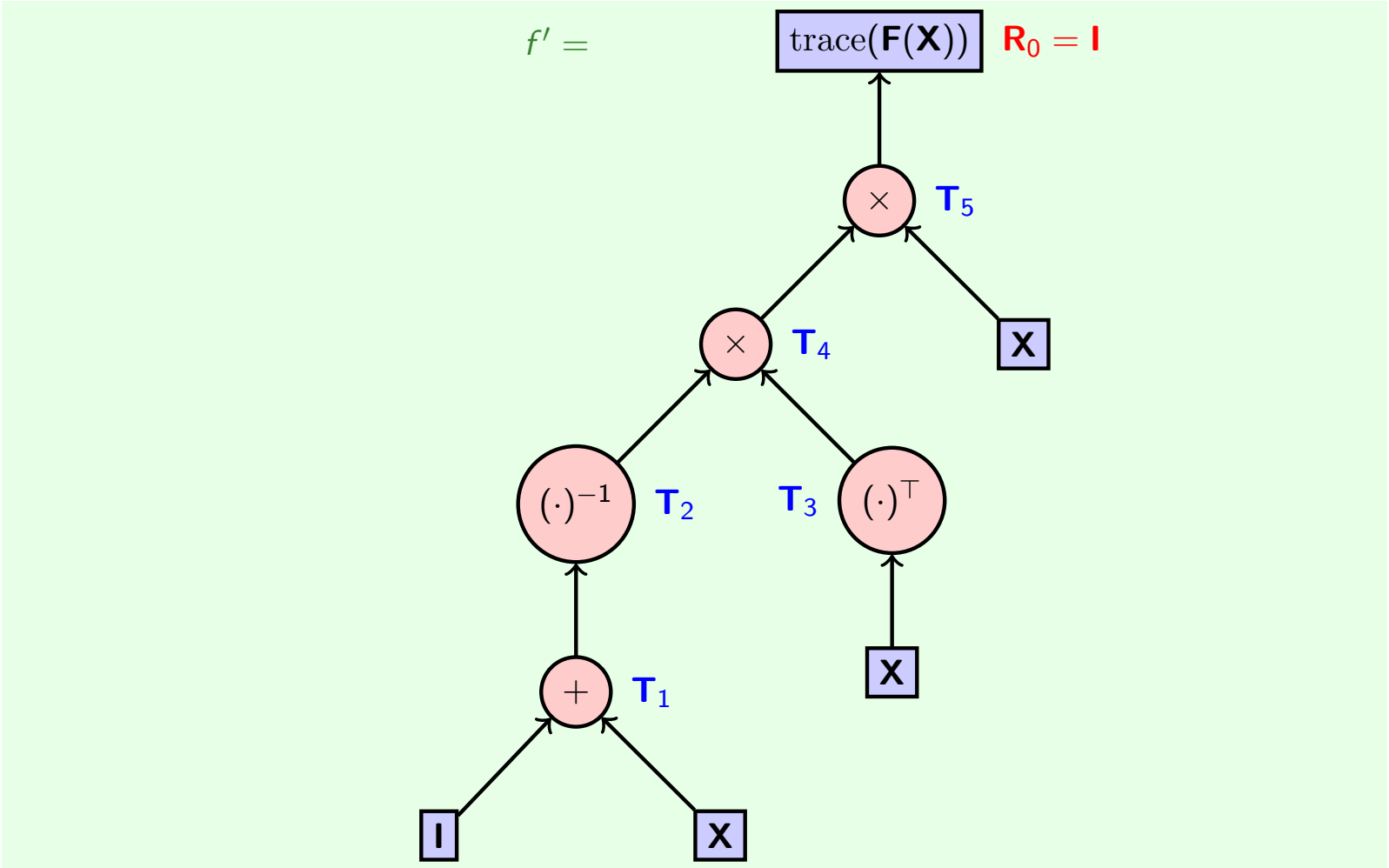
- $\mathbf{F}'(\mathbf{X})$ is composed of Kronecker, box (and outer) products for a large class of functions. (wow!)
- These can be "unwound" by multiplication with vectorized scalar-matrix derivatives (gasp!):

$$\mathrm{vec}^\top(\mathbf{C})\mathbf{A} \otimes \mathbf{B} = \mathrm{vec}^\top(\mathbf{B}^\top \mathbf{C}\mathbf{A})$$

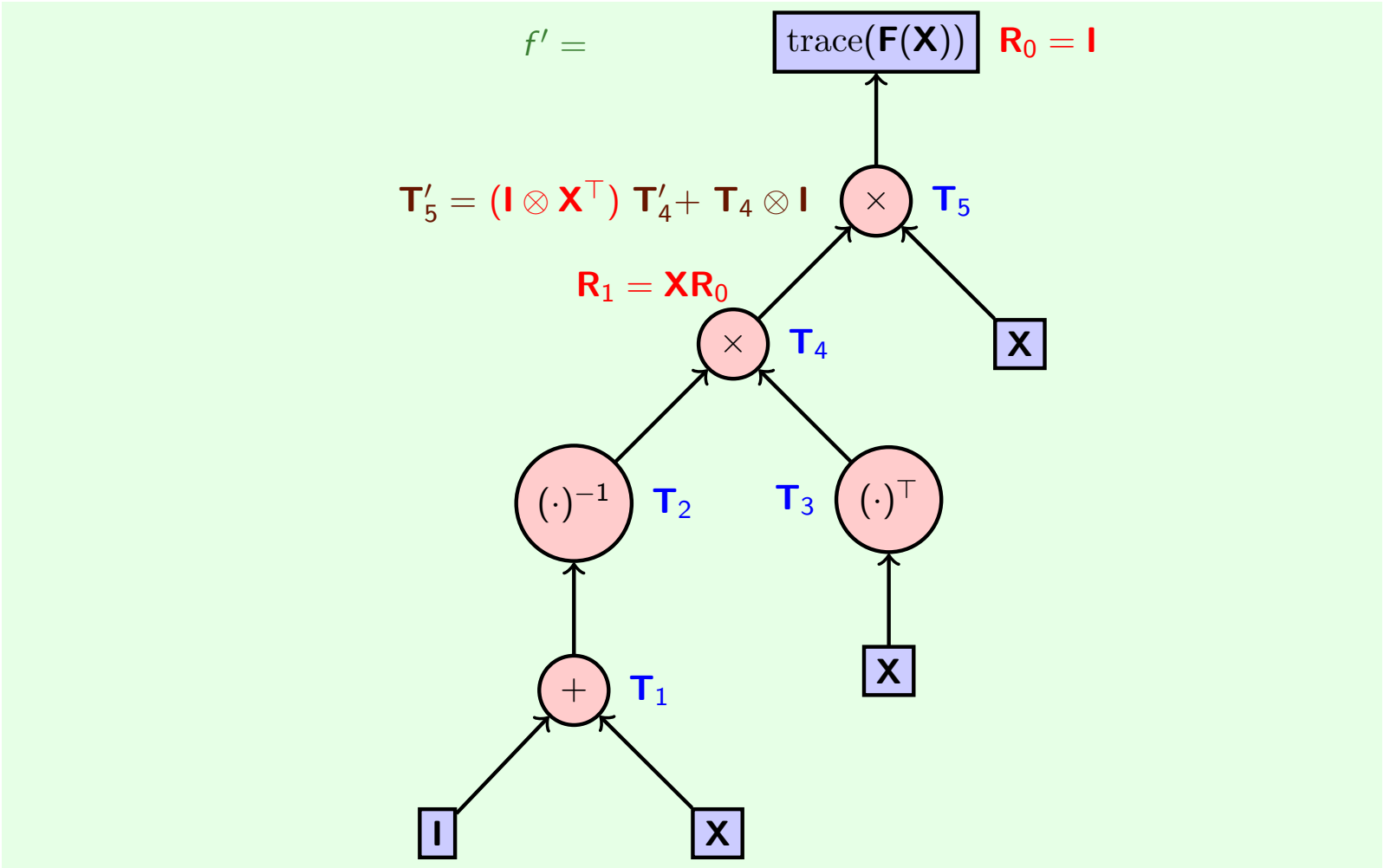
Reverse mode differentiation:

- Evaluate the derivative from top to bottom.
- Small scalar-matrix derivatives are propagated down the tree.

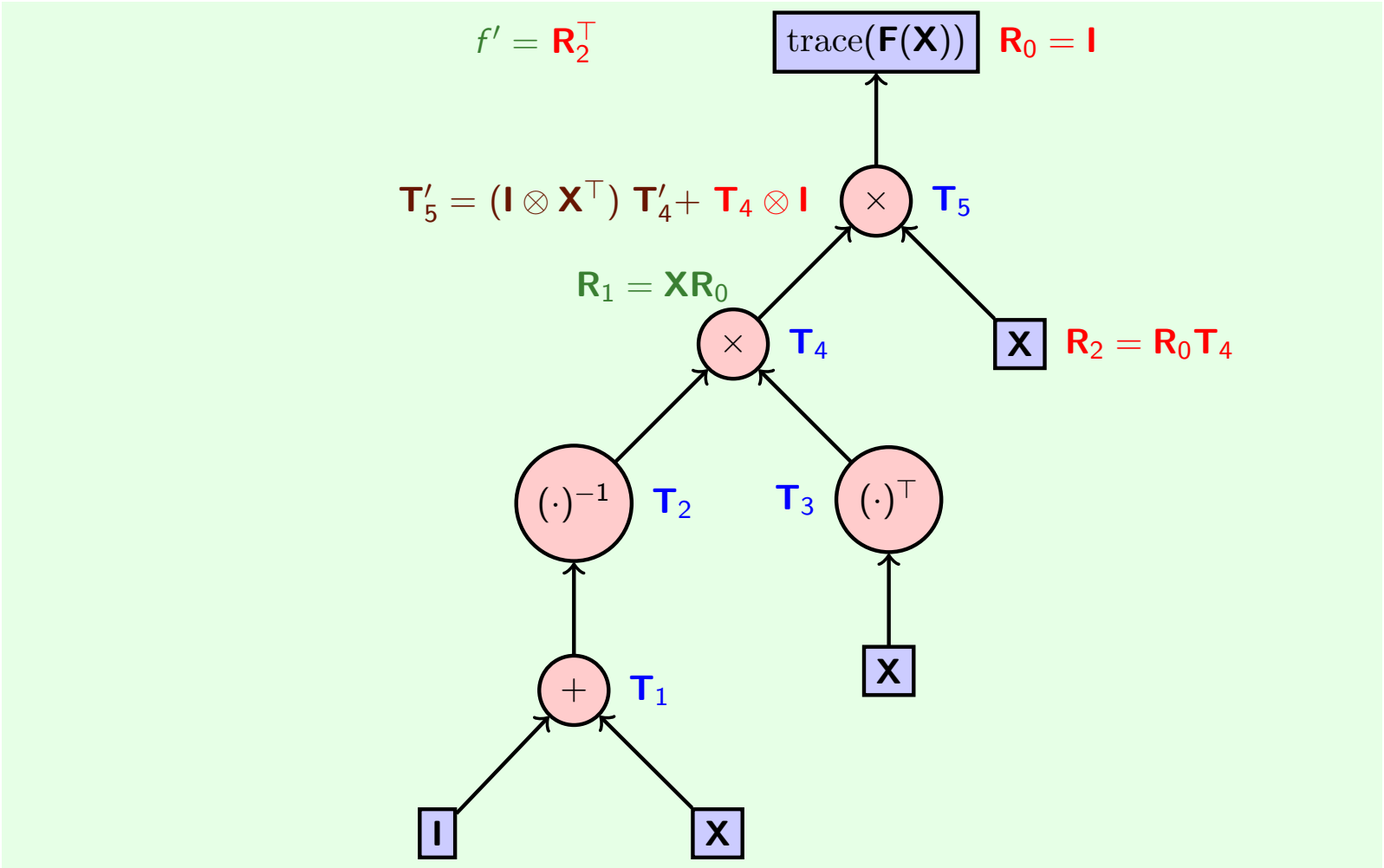
Reverse Mode Differentiation: $(f')^\top = \mathrm{vec}^{-1}\left(\mathrm{vec}^\top(\mathbf{I})\mathbf{F}'\right)$



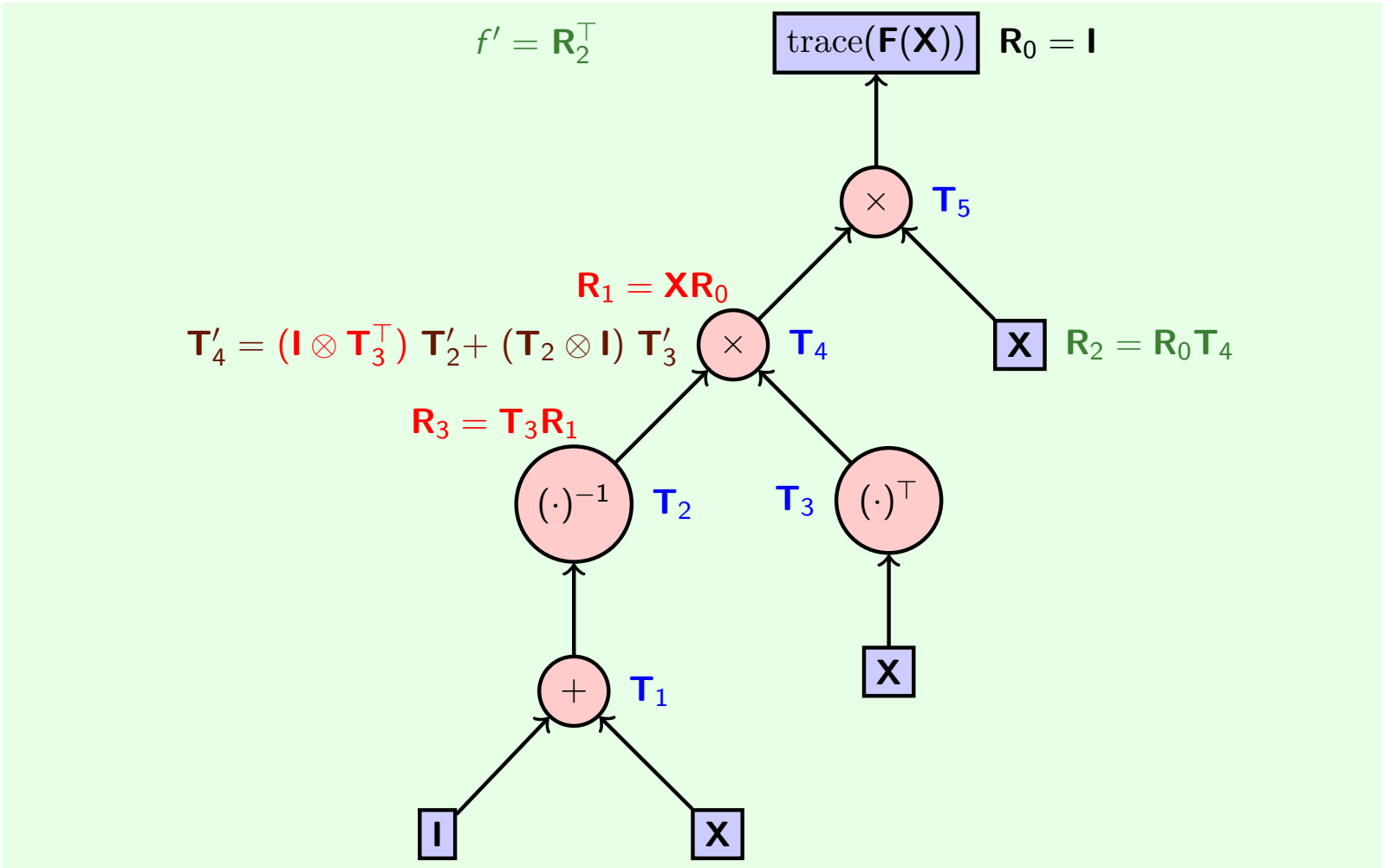
$$\text{vec}^\top(\mathbf{R}_0)(\mathbf{I} \otimes \mathbf{X}^\top) = \text{vec}^\top(\mathbf{X}\mathbf{R}_0\mathbf{I})$$



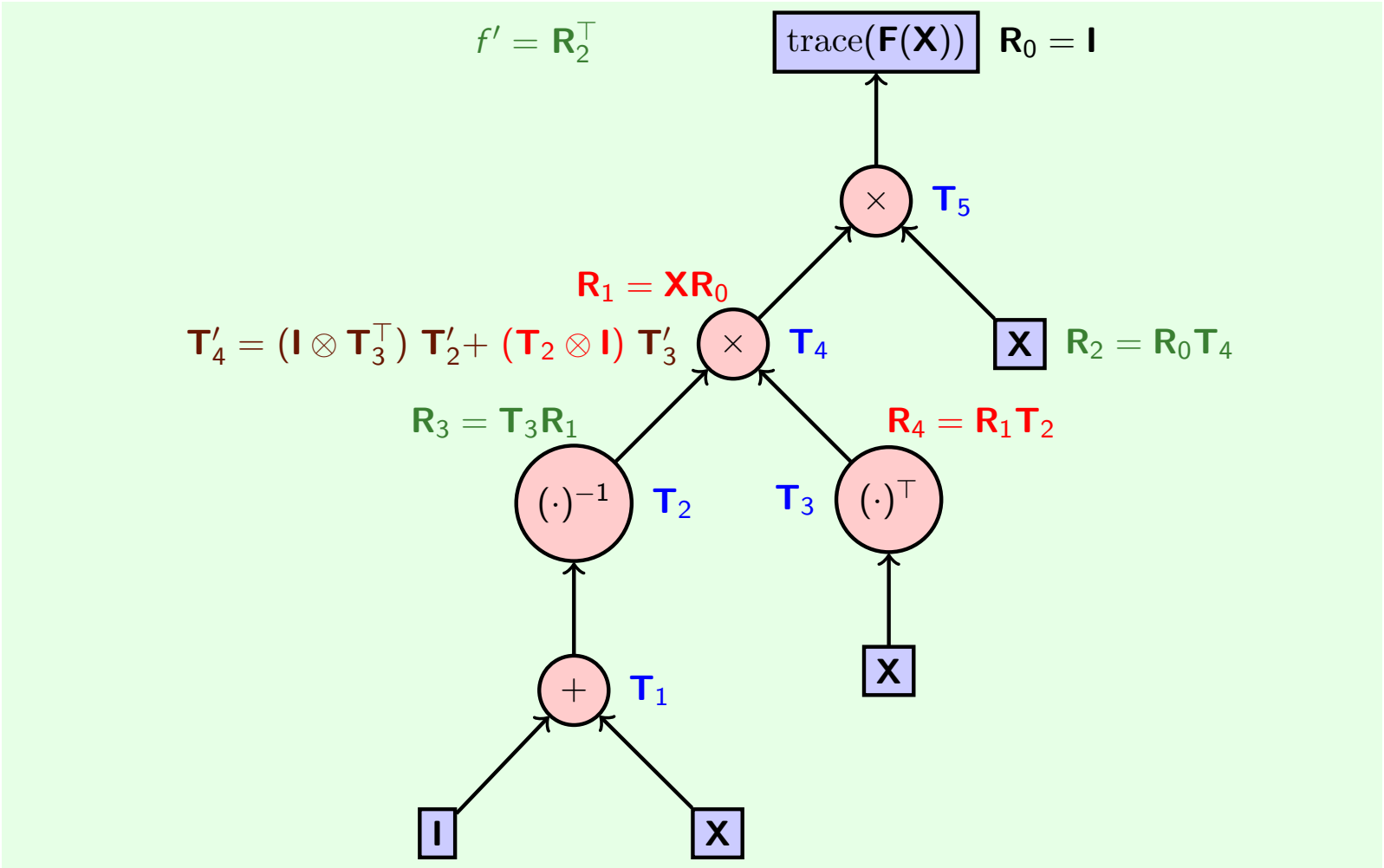
$$\mathrm{vec}^\top(\mathbf{R}_0)(\mathbf{T}_4 \otimes \mathbf{I}) = \mathrm{vec}^\top(\mathbf{I}\mathbf{R}_0\mathbf{T}_4)$$



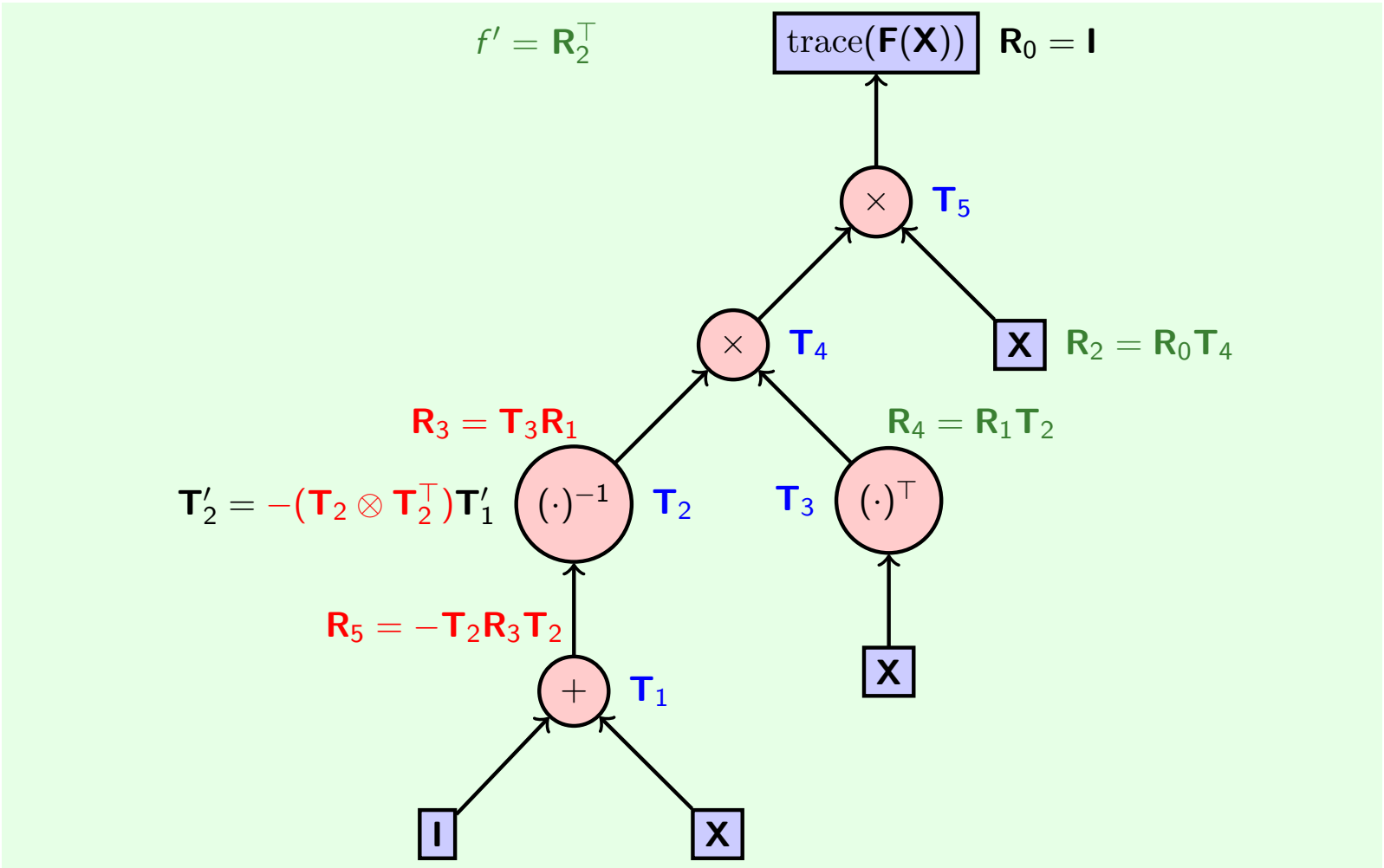
$$\text{vec}^\top(\mathbf{R}_1)(\mathbf{I} \otimes \mathbf{T}_3^\top) = \text{vec}^\top(\mathbf{T}_3\mathbf{R}_1\mathbf{I})$$



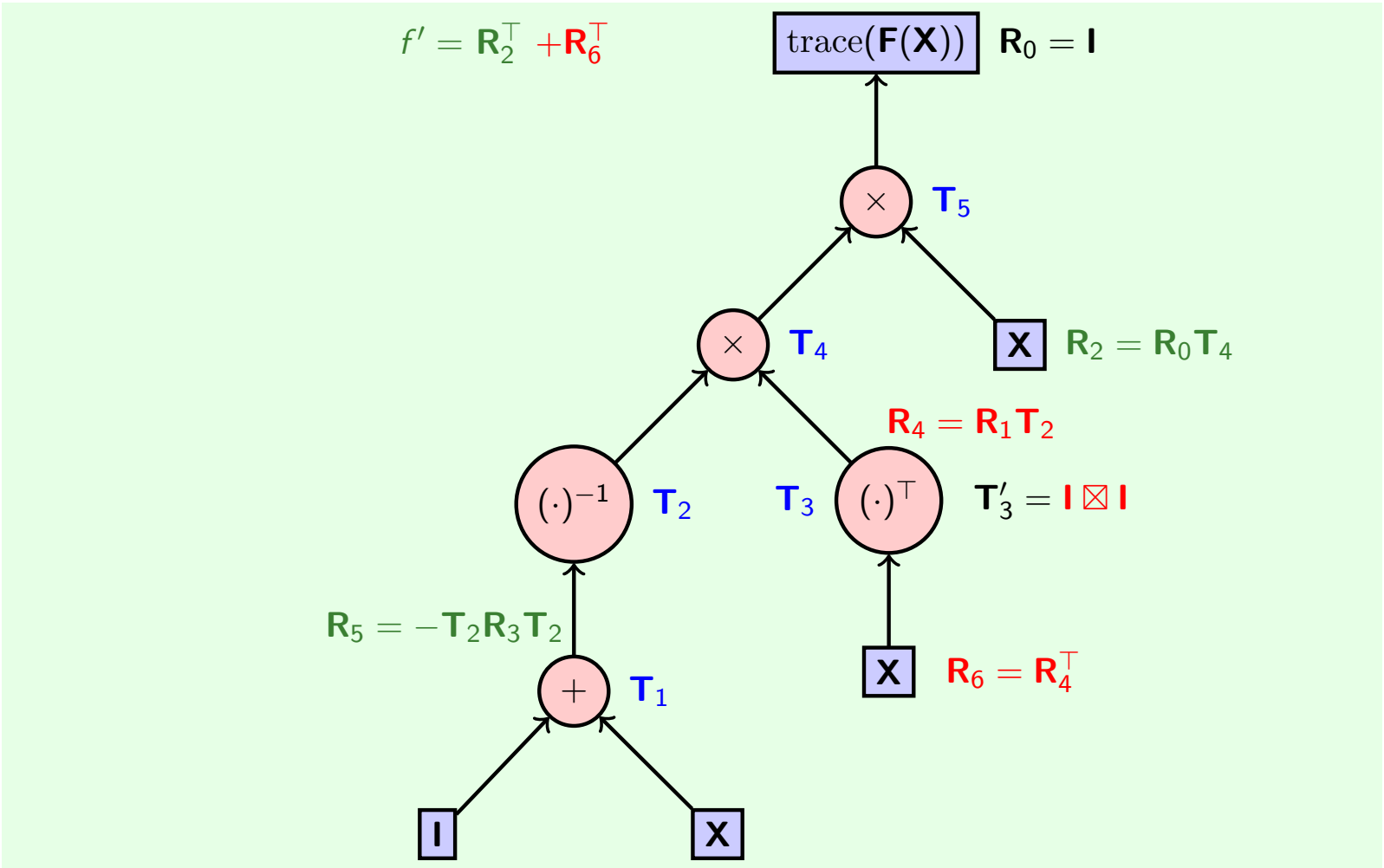
$$\text{vec}^\top(\mathbf{R}_1)(\mathbf{T}_2 \otimes \mathbf{I}) = \text{vec}^\top(\mathbf{I}\mathbf{R}_1\mathbf{T}_2)$$



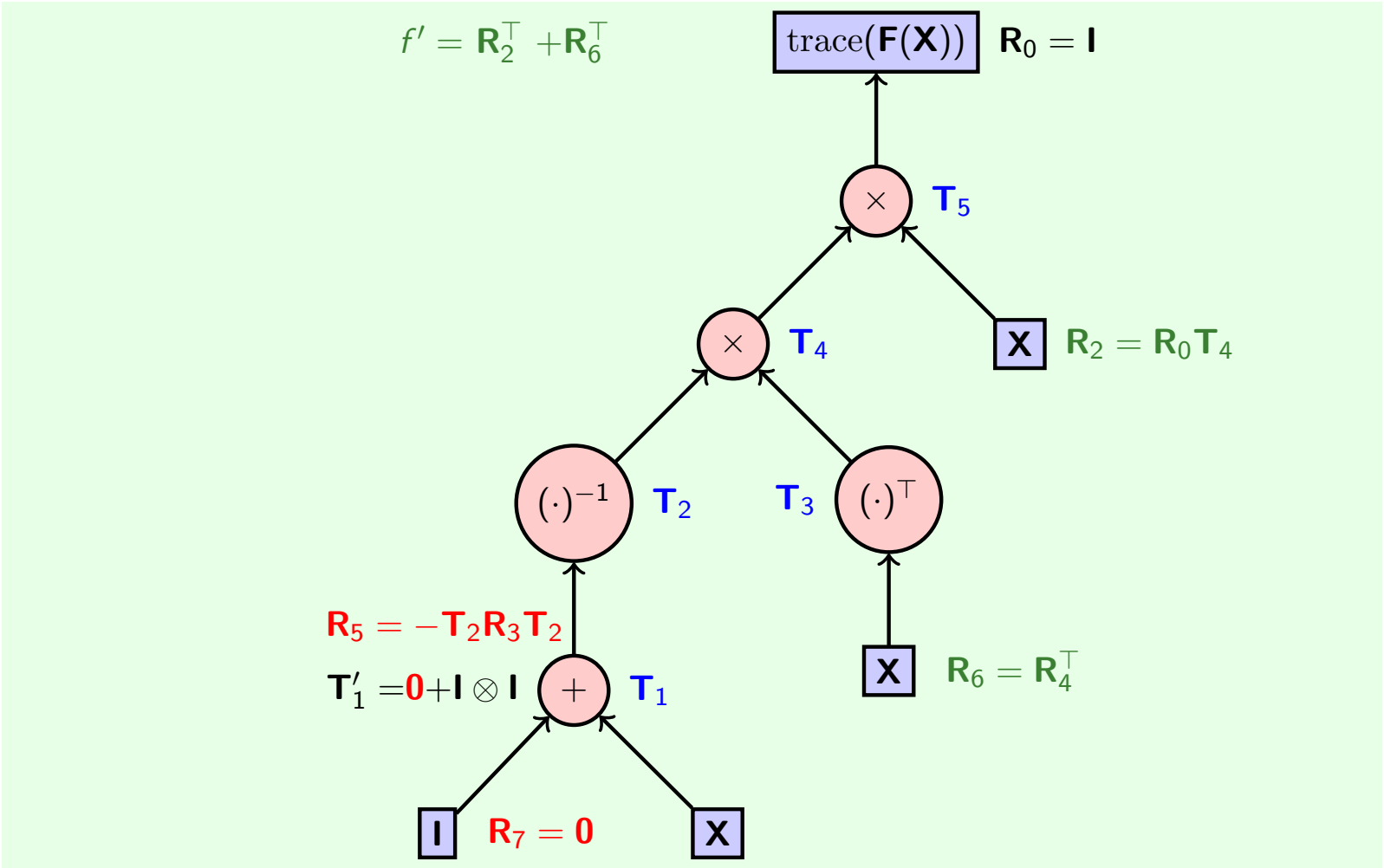
$$\text{vec}^\top(\mathbf{R}_3)(-\mathbf{T}_2 \otimes \mathbf{T}_2^\top) = \text{vec}^\top(-\mathbf{T}_2\mathbf{R}_3\mathbf{T}_2)$$



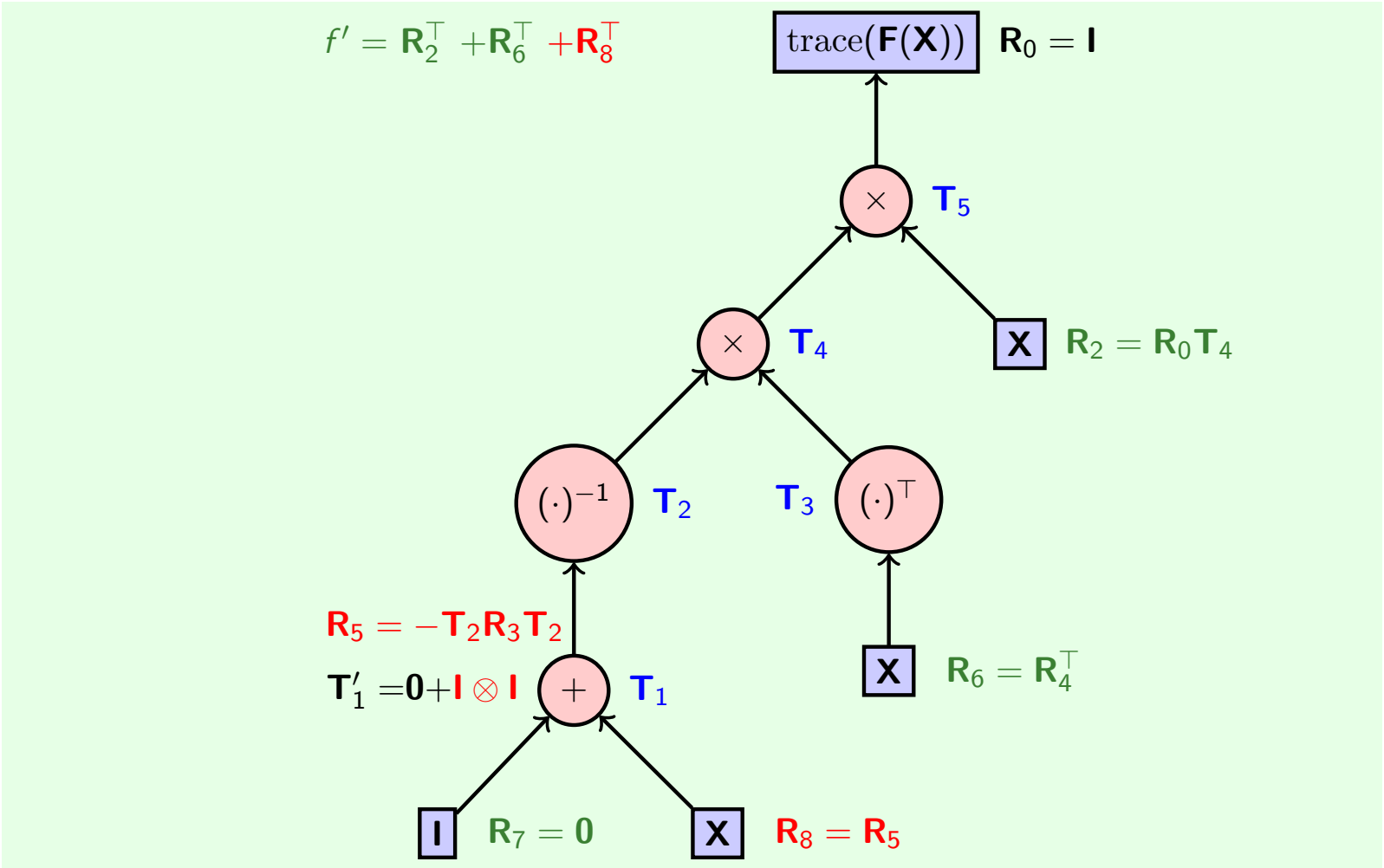
$$\text{vec}^\top(\mathbf{R}_4)(\mathbf{I} \boxtimes \mathbf{I}) = \text{vec}^\top(\mathbf{I}\mathbf{R}_4^\top \mathbf{I})$$



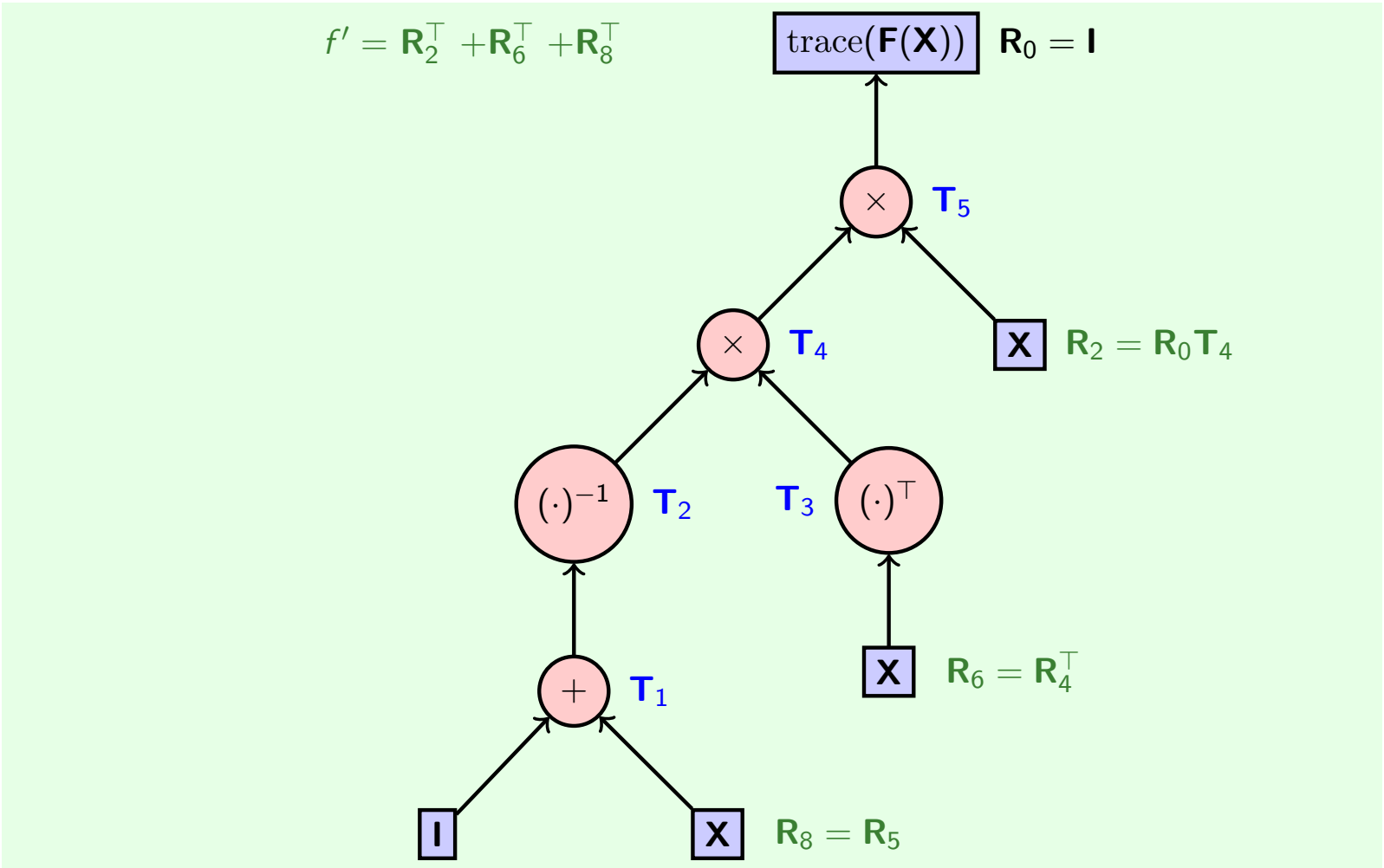
$$\text{vec}^\top(\mathbf{R}_5)\mathbf{0} = \text{vec}(\mathbf{0})$$



$$\mathrm{vec}^\top(\mathbf{R}_5)\mathbf{I} \otimes \mathbf{I} = \mathrm{vec}(\mathbf{R}_5)$$



$$f'(\mathbf{X}) = \mathbf{R}_2^\top + \mathbf{R}_6^\top + \mathbf{R}_8^\top$$



There is growing interest in Machine Learning in scalar–matrix objective functions.

- Probabilistic Graphical Models
  - Covariance Selection
- Optimization in Graphs and Networks.
  - Data-mining in social networks

Can we use this theory to help optimize such functions? We are on the look-out for interesting problems.

The Anatomy of a Matrix-Matrix Function

Theorem

Let $\mathbf{R}(\mathbf{X})$ *be a rational matrix–matrix function formed from constant matrices and* K *occurrences of* $\mathbf{X}$ *using arithmetic matrix operations* ($+$, $-$, $*$ *and* $(\cdot)^{-1}$) *and transposition* ($(\cdot)^\top$). *Then the derivative of the matrix–matrix function is of the form*

$$\sum_{i=1}^{k_1} \mathbf{A}_i \otimes \mathbf{B}_i + \sum_{i=k_1+1}^{K} \mathbf{A}_i \boxtimes \mathbf{B}_i.$$

The matrices $\mathbf{A}_i$ *and* $\mathbf{B}_i$ *are computed as parts of* $\mathbf{R}(\mathbf{X})$.

Hessian Forms

The derivative of a function f of the form $\mathrm{trace}(\mathbf{R}(\mathbf{X}))$ or $\log\det(\mathbf{R}(\mathbf{X}))$ is a rational function. Therefore, the Hessian is of the form

$$\sum_{i=1}^{k_1} \mathbf{A}_i \otimes \mathbf{B}_i + \sum_{i=k_1+1}^{K} \mathbf{A}_i \boxtimes \mathbf{B}_i.$$

where K is the number of times $\mathbf{X}$ occurs in the expression for the derivative.

If $\mathbf{A}_i$, $\mathbf{B}_i$ are $d \times d$ matrices then multiplication by $\mathbf{H}$ can be done with $\mathcal{O}(Kd^3)$ operations.

Generalizations of this result can be found in the paper.

Let $f : \mathbb{R}^{d \times d} \to \mathbb{R}$ with $f(\mathbf{X}) = \mathrm{trace}(\mathbf{R}(\mathbf{X}))$ or $f(\mathbf{X}) = \log\det(\mathbf{R}(\mathbf{X}))$.

- Derivative: $\mathbf{G} = f'(\mathbf{X}) \in \mathbb{R}^{d \times d}$ and Hessian: $\mathbf{H} = f''(\mathbf{X}) \in \mathbb{R}^{d^2 \times d^2}$.
- Newton direction $\mathrm{vec}(\mathbf{V}) = \mathbf{H}^{-1}\mathrm{vec}(\mathbf{G})$ can be computed efficiently if $K \ll d$

K	Algorithm	Complexity	Storage
1	$\mathbf{A} \otimes \mathbf{B} = \mathbf{A}^{-1} \otimes \mathbf{B}^{-1}$	$\mathcal{O}(d^3)$	$\mathcal{O}(Kd^2)$
2	Barthels-Stewart algorithm	$\mathcal{O}(d^3)$	$\mathcal{O}(Kd^2)$
≥ 3	Conjugate Gradient Algorithm	$\mathcal{O}(Kd^5)$	$\mathcal{O}(Kd^2)$
$\geq d$	General matrix inversion	$\mathcal{O}(d^6 + Kd^4)$	$\mathcal{O}(d^4)$

The (generalized) Bartels-Stewart algorithms solves the Sylvester-like equation $\mathbf{AX} + \mathbf{X}^\top\mathbf{B} = \mathbf{C}$.

What about optimizing functions of the form

$$f(\mathbf{X}) + \|\mathrm{vec}(\mathbf{X})\|_1?$$

Common approaches:

Strategy	method	Use Hessian structure?
Newton-Lasso	Coordinate descent	✓
	FISTA	✓
Orthantwise ℓ_1	CG	✓
	L-BFGS	✗

It is not obvious how to take advantage of the Hessian structure for all these methods. For orthantwise CG for example – the sub-problem requires the Newton direction for a sub-matrix of the Hessian.

1. We have applied this methodology to the covariance selection problem – a paper is forthcoming.
2. We are digging deeper in the theory of matrix differentiation and properties of the box products.
3. We are on the look-out for more interesting matrix optimization problems. All suggestions are appreciated!