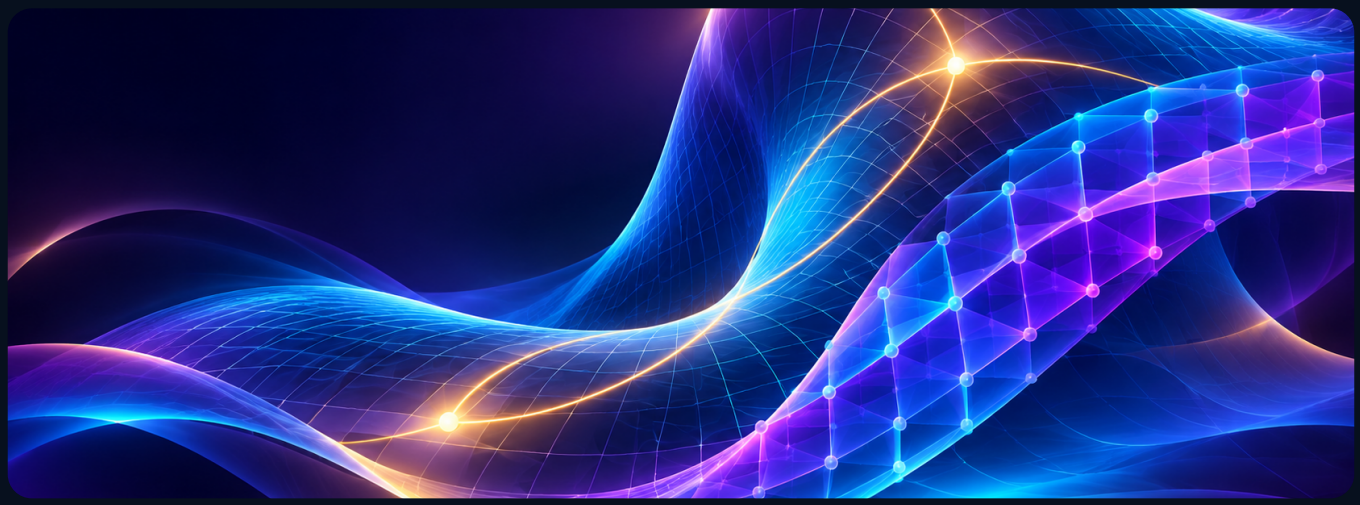
SYMBRAIX / USER GUIDE

Mathematics, always at hand.

A beautifully minimal scientific scratchpad for numerical, symbolic, and indexed tensor work.



Version 0.7.0 (8) · November 2026 · Dark edition
Symbraix for macOS · Documentation for the internal beta

CONTENTS

Explore the guide

From a single expression to metric-aware tensor work, with a record of what each example demonstrates.

01	A small window for serious mathematics
02	Your first calculation
03	Scientific notation and variables
04	Vectors, complex numbers and matrices
05	Quantities that know what they mean
06	A practical layer of algebra
07	From expression to picture
08	Tensor values and index metadata
09	From Hamiltonian structure to complex variables
10	Barred indices are geometric information
11	Gauge kinematics from the dissertation
12	A metric on the sphere
13	A damped control response
14	Dynamics in complex phase space
15	Entropy in one line
16	Know the limits of a result
17	Settings, appearance and saved work
18	Sources and reproducibility

The scientific charts are analytic illustrations. Image-specific calculations are described in their chapters. Links and build status may change before public release.

01 / WHY WE BUILT IT

A small window for serious mathematics

We created Symbraix because a quick calculation should not require opening a programming environment, loading packages, or setting up an analysis notebook. We wanted mathematical depth without the overhead.

Symbraix stays alongside the work. A single expression can evaluate a trigonometric identity or a complex number; the same workspace can hold vectors, matrices, uncertainties, symbolic algebra, plots, and numerical tensor operations. We can revisit an earlier result without losing the context in which it was obtained.

Our ambition is broader than a calculator, but our interface remains deliberately modest. Symbraix is a focused scientific scratchpad for researchers, engineers, students, and anyone who thinks in equations.

> **Minimal to operate. Capable enough to explore. Transparent about what has and has not been validated.**



Concept artwork: a design illustration of the intended workflow, not a screenshot of the current TestFlight build.

Your first calculation

Open Symbraix. Type an expression in the lower editor, then press **Return**. Every successful statement becomes an item in the scrollable worksheet above.

```
vec(1,2,3,4) + vec(1,2,3,4)
```

Expected result or note: Result: [2, 4, 6, 8]

Definitions and evaluations can share one submission. Statements are separated by semicolons or new lines; semicolons inside square brackets remain matrix row separators.

```
x=4; y=7; x+y**2
```

Expected result or note: Results: 4, 7, 53

Shift-Return inserts a line without evaluating. **Up/Down** recall earlier commands at editor boundaries, while **Option-Up/Option-Down** explicitly cycle command history. Existing results are labelled `Out[n]`; `out(n)` reuses their full stored values.

Control	Purpose
New / Command-N	Begin a fresh calculation worksheet
Open / Command-O	Open a previously saved chain file
Save / Command-S	Save the current chain, including results
Settings / Command-,	Set input and output text sizes independently
Keep on Top	Keep the calculator visible beside other apps

The examples in this guide use a readable input syntax and are not a promise of arbitrary Python compatibility.

Scientific notation and variables

Expressions are deterministic. Names store results, not live formulas; later reassignment does not rewrite old entries. Built-in functions use radians unless they explicitly specify degrees.

```
q=2; p=3
z=(q+i*p)/sqrt(2)
z*conj(z)
```

Expected result or note: Result: 6.5

Notation	Interpretation
^ and **	Exponentiation; 2^3^2 evaluates as 512
i or j	Imaginary unit, i*i = -1
pi, e, tau	Mathematical constants
sin, cos, tan, exp	Elementwise numerical functions
ln, log2, log_10	Logarithms by base
ans	Most recent successful result
out(n)	Saved full-precision worksheet result

A malformed statement leaves its input available for correction. The worksheet and variable state of that submission are rolled back together, avoiding partially applied calculations.

Precision: ordinary real and complex numerical values use 64-bit floating point. Extra printed digits do not imply extra accuracy.

Vectors, complex numbers and matrices

Array operations follow a NumPy-like distinction: `*` is elementwise multiplication and `@` is matrix multiplication. The transpose is not the complex adjoint.

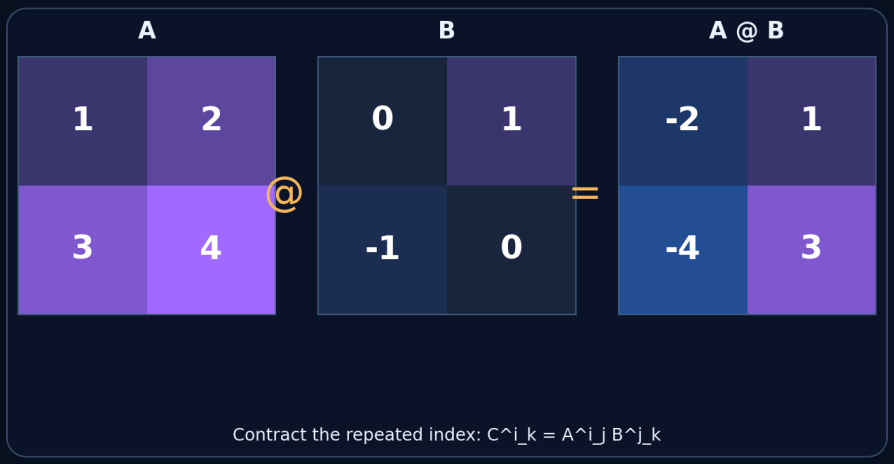
```
A=[1,2;3,4]
A*A
A@A
det(A)
inv(A)
```

Expected result or note: [[1,4],[9,16]] · [[7,10],[15,22]] · -2 · [[-2,1],[1.5,-0.5]]

Complex operations extend through vectors and matrices, including multiplication, conjugation and supported scientific functions. For inner products, `dot(a,b)` is bilinear, whereas `vdot(a,b)` conjugates its first argument.

```
u=[1+i,2]
vdot(u,u)
transpose(A)
adjoint(A)
```

Expected result or note: 6 · ordinary transpose · conjugate transpose



Matrix multiplication as an explicit contraction, not elementwise arithmetic.

Some numerical factorization operations have size limits. The algebra engine is not a general matrix-function or arbitrary-precision package.

05 / UNITS & ERRORS

Quantities that know what they mean

Dimensional compatibility is checked when quantities are added or converted. Measurements carry standard uncertainties through supported operations using first-order derivative propagation.

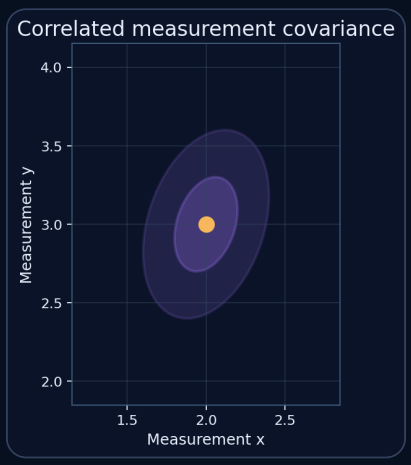
```
length=measure(2,0.01,"m")
width=measure(30,0.2,"cm")
convert(length*width,"m^2")
```

Expected result or note: 0.6 +/- 0.005 m^2

Reusing a measurement retains its identity and therefore its correlation. Two separately constructed measurements are independent unless an explicit correlation is provided.

```
a=measure(2,0.1,"m")
sigma(a-a)
sigma(a+a)
```

Expected result or note: 0 · 0.2



Uncertainty is a joint geometric object when input errors are correlated.

First-order propagation is an approximation, especially near singularities or for strongly nonlinear functions. This version does not provide general uncertain tensor or Monte Carlo calculations.

06 / SYMBOLIC

A practical layer of algebra

Symbolic expressions are constructed explicitly. This avoids confusing a misspelled numerical name with a new mathematical indeterminate. The supported manipulations are intentionally conservative.

```
simplify("2*x+3*x")
expand("(x+1)^2")
differentiate("x^3+2*x","x")
evaluate("x^2+1","x",[1,2,3])
```

Expected result or note: 5*x · expanded polynomial · 3*x^2+2 · [2,5,10]

The symbolic system handles a defined subset of simplification, expansion, substitution, differentiation and low-degree polynomial analysis. It does not automatically prove identities that depend on complex branches, positive-definiteness, or unspoken assumptions.

> **Treat an algebraic rewrite as a transformation with hypotheses, not a cosmetic simplification.**

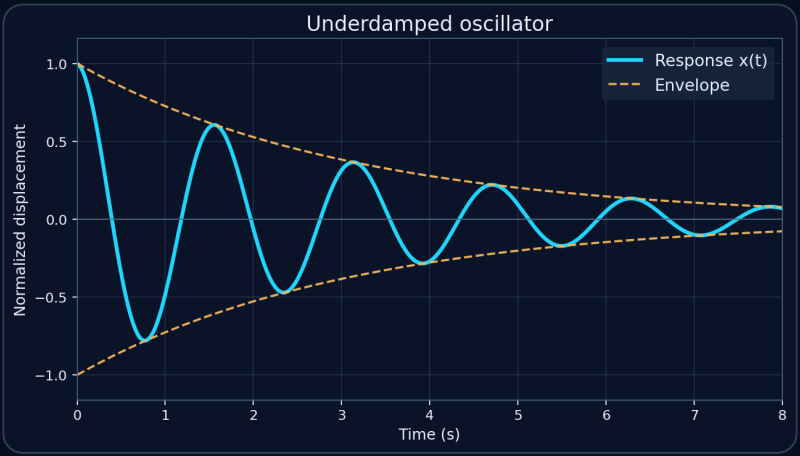
General-purpose symbolic tensor canonicalization, arbitrary symbolic integration and metric variation remain development goals rather than capabilities of the current beta.

07 / PLOTTING

From expression to picture

Define data in the same worksheet, then plot it directly. The plot captures the data as they were calculated; later changes to a variable do not alter the earlier result.

```
t=linspace(0,8,401)
omega=4; zeta=0.08
x=exp(-zeta*omega*t)*cos(omega*sqrt(1-zeta^2)*t)
plot(t,x,"Damped response")
```



An illustrative damped oscillator with its analytical envelope. See the engineering example.

Line/scatter display, multiple curves and SVG figure export are supported in the underlying beta source. Plotting is sampled, not guaranteed to detect every discontinuity between samples.

Tensor values and index metadata

Array rank alone is not enough for geometric tensor algebra. Our typed numerical tensor layer records index variance (upper/lower), conjugation kind (barred/unbarred), and a named carrier space.

Text notation	Interpretation
^a	Upper, unbarred index
_a	Lower, unbarred index
^~a	Upper, barred index
_~a	Lower, barred index

```
h=metric([1,0;0,1],"V","hermitian")
u=indexed([1+i,2],"^a","V")
tinner(u,u,h)
```

Expected result or note: 6

Other operations include component extraction, slot permutations, transposes, raising/lowering with explicit forms, compatible contraction, and numerical frame changes. Barred indices are not merely a synonym for covariant indices.

This layer checks numerical indexed components. It does not yet automatically perform abstract dummy-index renaming, Bianchi canonicalization, covariant derivative identities, or full symbolic curvature calculus.

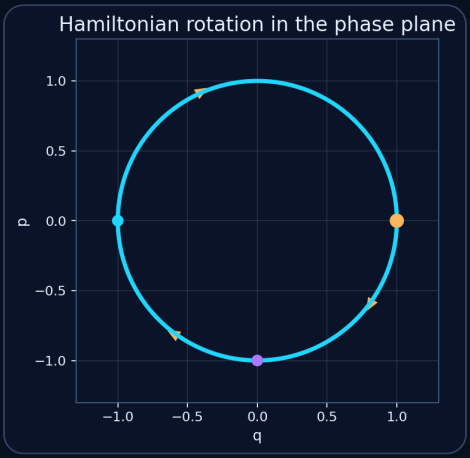
09 / SOURCE EXAMPLE

From Hamiltonian structure to complex variables

Dean (2026), *Hamilton's equations in unitary form*, Eq. (8), begins with the antisymmetric 2×2 tensor that organizes a coordinate-momentum pair. A direct numerical check is:

```
J=[0,1;-1,0]
J@J
J@[2,3]
```

Expected result or note: [[-1,0],[0,-1]] · [3,-2]



For H=(q^2+p^2)/2, the flow generated by J traces an orbit in phase space.

The matrix identity $J^2 = -I$ explains the appearance of eigenvalues ±i. The manuscript's Eq. (13) defines complex variables from q and p; an executable scalar version is `z=(q+i*p)/sqrt(2)`.

Source: Dean (2026), pp. 4–6, Eqs. (8), (11)–(16). Numerical checks supported; full symbolic derivation not claimed.

10 / SCHOUTEN NOTATION

Barred indices are geometric information

Dean (2026), p. 5, draws on J. A. Schouten's hybrid-index notation to place conjugation on the index rather than only the component symbol. This is distinct from upper-versus-lower variance. Symbraix therefore records these as separate slot attributes.

```
h=metric([1,0;0,1],"V","hermitian")
u=indexed([1+i,2],"^a","V")
tinner(u,u,h)
```

Expected result or note: 6

The example above illustrates a standard positive Hermitian form. It is **not** an assertion that the special transformed tensor in the manuscript's Eq. (14) is identical to the identity Hermitian form.

Convention check for research replication: Eq. (14) of the supplied manuscript displays diagonal entries −i and +i while calling the displayed matrix Hermitian. Under the usual conjugate-transpose convention, that diagonal matrix is skew-Hermitian. Symbraix should retain the source exactly, record the convention question, and avoid silently changing signs.

Sources: Dean (2026), pp. 5–6, Eqs. (13)–(16); Schouten (1954), as cited in that article.

Gauge kinematics from the dissertation

Dean's 1999 dissertation, Appendix C, develops a rotational gauge connection and its relation to motion in a rotating frame. The printed connection matrix in Eq. (C.9) is antisymmetric.

For a particular angular-velocity choice, the following is a straightforward component test following the sign convention displayed in that equation:

```
W=[0,3,-2;-3,0,1;2,-1,0]
transpose(W)+W
W@[1,0,0]
```

Expected result or note: Zero matrix · [0,-3,2]

This verifies antisymmetry and matrix-vector action, not the gauge-transformation law or a variational identity. Reproducing Eq. (C.10), the covariant derivative and full local SO(3) kinematics symbolically would require the extended tensor-field engine.

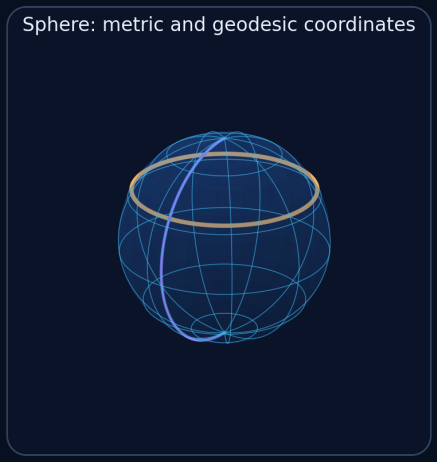
Source: Dean (1999), Appendix C, pp. 166–168, Eqs. (C.5)–(C.13). Numerical component adaptation.

A metric on the sphere

On a unit sphere with coordinates (θ, φ), the standard two-dimensional metric is `diag(1, sin²θ)`. Symbraix can evaluate its numerical components and inverse at a selected point.

```
theta=pi/3
g=[1,0;0,sin(theta)^2]
det(g)
inv(g)
```

Expected result or note: 0.75 · [[1,0],[0,1.333333...]]



Metric coordinates on a sphere: latitude/longitude directions have different length factors.

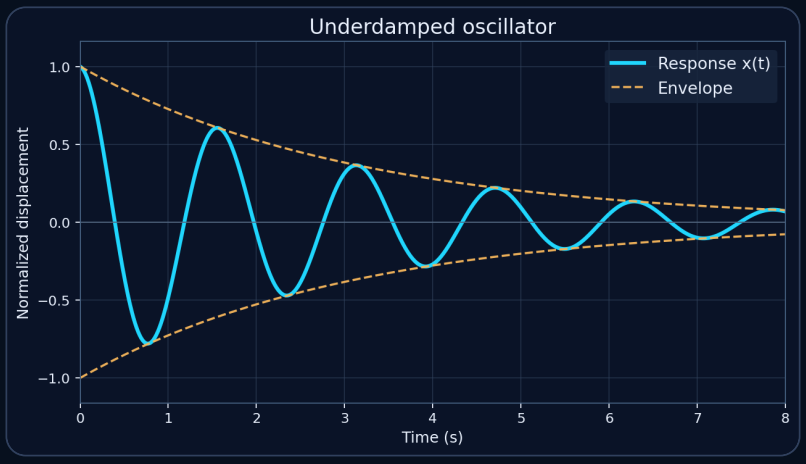
For a sphere of radius R, the Gaussian curvature $K = 1/R^2$ is a mathematical reference result. Automatically deriving the Christoffel symbols and curvature from an arbitrary symbolic metric is **not** a verified Build 8 feature.

Reference: Standard sphere metric. The related geometric motivation appears in Dean (1999), Chs. 2–4; this specific sphere example is an independently authored teaching example.

A damped control response

A second-order underdamped system has a response with an exponentially decreasing envelope and a damped natural frequency. A vectorized workflow makes it easy to inspect its sensitivity to parameters.

```
omega=4; zeta=0.08
t=linspace(0,8,401)
wd=omega*sqrt(1-zeta^2)
x=exp(-zeta*omega*t)*cos(wd*t)
plot(t,x,"Damped response")
```



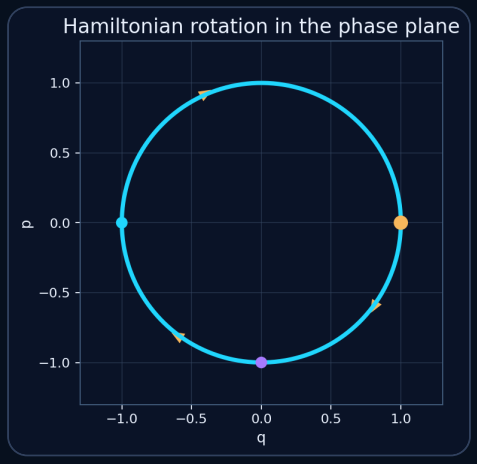
Analytical damped response. Natural frequency 4 rad/s; damping ratio 0.08.

Unit-aware quantities and correlated measurements can be used separately for calibration. A complete symbolic transfer-function or robust-control toolbox is outside this initial beta.

Dynamics in complex phase space

The complex coordinate z = (q+ip)/√2 in Dean (2026), Eq. (13), joins a real phase-space pair for a chosen normalization. For a harmonic oscillator, the motion takes the form z(t)=z(0)e^(−iωt) in Eq. (28).

```
omega=2; t=linspace(0,2*pi,501)
z0=1+0.5*i
z=z0*exp(-i*omega*t)
plot(real(z),imag(z),"Complex trajectory")
```



Real phase-plane orbit for an ideal oscillator; the complex trajectory is the same motion in another representation.

This example exercises complex vectorization and 2D plotting. It does not, by itself, establish invariance under arbitrary canonical transformations or symbolic Poisson-bracket identities.

Source: Dean (2026), Section VI, pp. 8–9, Eqs. (25)–(29).

Entropy in one line

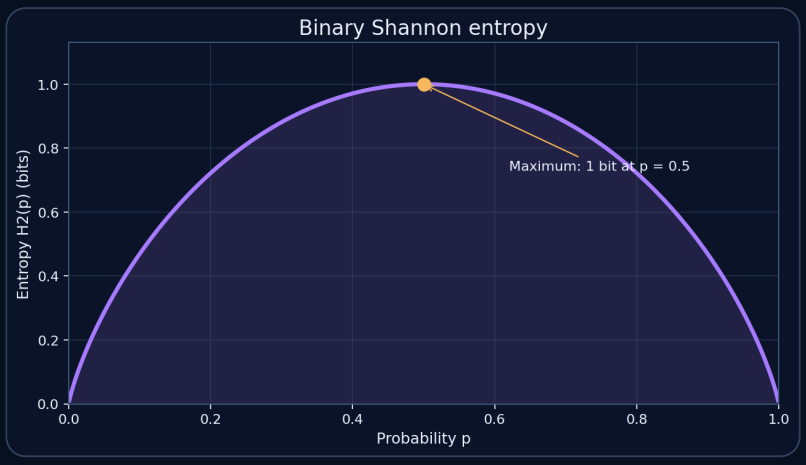
The binary Shannon entropy measures uncertainty in a Bernoulli variable with probability p, in bits:

$$H_2(p) = -p \log_2 p - (1-p) \log_2(1-p)$$

The calculator can evaluate the expression elementwise. Avoid p=0 or 1 with direct logarithms and evaluate the limiting values separately.

```
p=linspace(0.01,0.99,199)
H=-(p*log2(p)+(1-p)*log2(1-p))
plot(p,H,"Binary entropy")
```

Expected result or note: At p=0.5, H = 1 bit



Binary entropy peaks at p = 1/2, where $H_2 = 1$ bit.

This is a useful example of mathematical depth without a specialized statistics or communications interface. More elaborate channel-coding and inference algorithms are outside the release scope.

Reference: Shannon (1948), *A Mathematical Theory of Communication*, Bell System Technical Journal.

16 / NUMERICAL INTEGRITY

Know the limits of a result

Symbraix uses standard floating-point computations for most numerical values. It aims for deterministic and explainable behavior, but it is not a certified arbitrary-precision or interval-arithmetic system.

Area	Current boundary
Floating point	64-bit real and complex values
Vector/matrix/tensor	Finite numerical arrays with shape and resource limits
Symbolic manipulation	Restricted algebra; no full CAS assumptions engine
Tensor algebra	Index-aware components; incomplete abstract tensor calculus
Uncertainty	First-order scalar error propagation
Plots	Sampled two-dimensional data; no adaptive guarantee

Retain the source expression, units, coordinate conventions, and error assumptions for any result used in design, publication, or safety-related work. Numerical comparisons should use stated tolerances, not equality of displayed decimal strings.

Important research boundary: the supplied dissertation's MathTensor workflows for curvature variations, Palatini equations and complete Einstein-Hilbert symbolic derivations are *not* verified as supported in this beta.

Settings, appearance and saved work

Symbraix follows the system light/dark appearance. In the updated development source, **Symbraix → Settings (Command-comma)** contains independent font-size settings for input and worksheet results. These were still awaiting fresh interactive acceptance when the first macOS TestFlight Build 8 was uploaded.

New begins a calculation; **Open** restores a saved chain without reevaluating; **Save** writes the state of the worksheet and supported result values. Explicit file operations should not change appearance preferences.

We plan to bundle matching **Light** and **Dark** PDF manuals. This documentation package provides both; linking them into the installed app requires updating and distributing the newer application bundle through the normal release process.

> A web account is not required to use the local Mac calculator. Community registration and support-form submission, once activated, are separate optional website services.

18 / SOURCES

Sources and reproducibility

Dean, Bruce H. (2026). *Hamilton's equations in unitary form*, manuscript dated September 17, 2026. Especially Secs. IV–VI and Eqs. (8), (11)–(16), (25)–(29). Supplied author manuscript.

Dean, Bruce Herold (1999). *Variational Results and Solutions in Gauge Gravity and a Bifurcation Analysis of Black Hole Orbital Dynamics*. Ph.D. dissertation, West Virginia University. Appendix A (MathTensor 2.2/Mathematica 3.0), Appendix C (gauge kinematics), Chapters 2–6.

Schouten, J. A. (1954). *Ricci-Calculus*, 2nd ed.; *Tensor Analysis for Physicists*, 2nd ed. Referenced in Dean (2026).

Christensen, S. M. and Parker, L. (1994). *MathTensor: A System for Doing Tensor Analysis by Computer*. Addison-Wesley. Referenced in Dean (1999).

Shannon, C. E. (1948). “A Mathematical Theory of Communication,” *Bell System Technical Journal*.

> **Examples adapted from the supplied papers are cited at the point of use. Independently authored educational examples are identified as such. Concept artwork does not imply that a pictured interface has been tested or distributed.**

Symbraix 0.7.0 (8) documentation edition, November 2026. This guide accompanies a beta release; features may change before public distribution.