ESHKOL — DESCRIPTION COPY

Eshkol is a compiled programming language for mathematical computing developed by Tsotchke Corporation. It is the first production language to integrate automatic differentiation—the mathematical technique that enables neural networks to learn—directly at the compiler level.

Automatic differentiation is foundational to modern machine learning. Every neural network trained today relies on it to compute gradients and update parameters. Yet in every major programming language, this capability exists as an external addition—libraries and frameworks layered onto systems never designed to support them. Eshkol takes a different approach. Differentiation is intrinsic to the language, as native as addition or multiplication.

The compiler provides three modes: symbolic differentiation performed entirely at compile-time with zero runtime cost, forward-mode through dual number arithmetic for efficient first derivatives, and reverse-mode through computational graph construction for backpropagation across functions with millions of parameters. Gradients can nest to arbitrary depth, enabling Hessians and higher-order derivatives for second-order optimization. Eight vector calculus operators—gradient, Jacobian, Hessian, divergence, curl, Laplacian, directional derivative, and standard derivative—exist as language primitives rather than library calls.

Eshkol eliminates garbage collection entirely. Memory is managed through arena-based allocation with ownership-aware lexical regions, providing deterministic performance with microsecond-scale worst-case guarantees. Allocation runs in constant time. Deallocation happens predictably at scope boundaries. For applications where timing cannot be left to chance—robotics, autonomous systems, real-time inference, safety-critical AI—this predictability is not optional.

The language descends from Scheme but compiles to native machine code through LLVM, targeting both x86-64 and ARM64 architectures. Unusually for a compiled language, it preserves homoiconicity—Lisp's property of representing code as manipulable data structures. Compiled closures retain their symbolic S-expression forms, accessible at runtime for introspection and transformation. No other compiled language maintains this capability.

A gradual type system grounded in Homotopy Type Theory supports dependent types for dimensional correctness and resource linearity, providing compile-time verification when desired while remaining unobtrusive during exploratory work.

Tsotchke Corporation built Eshkol as foundational infrastructure for its quantum-AI research, including Selene, a geometric large language model achieving significant efficiency gains through mathematical techniques Eshkol enables, and Moonlab, a high-performance quantum computing simulator. The language was created to solve problems no existing tool could address. It is now released to the wider research and engineering community under the MIT license.

Quote from Tyrell Edward Umbach, CEO and creator:

> "We built Eshkol because the future of computing requires a language that treats mathematical concepts—automatic differentiation, geometric operations, deterministic performance—as fundamental rather than afterthoughts. We're releasing it because researchers and engineers building the next generation of intelligent systems deserve tools designed for this era, not adapted from the last one."

Resources

- **Website**: https://eshkol.ai
- **Source**: https://github.com/tsotchke/eshkol

Word count: 428