Writing astronomy codes using GenAI

Yogesh Wadadekar

January 2025

NCRA • TIFR

before we begin - A quick poll

- How many of you have used generative AI for non-coding related tasks?

before we begin - A quick poll

- How many of you have used generative AI for non-coding related tasks?
- How many of you have used generative AI for coding?

before we begin - A quick poll

- How many of you have used generative AI for non-coding related tasks?
- How many of you have used generative AI for coding?
- Which programming language have you used with generative AI?

before we begin - A quick poll

- How many of you have used generative AI for non-coding related tasks?
- How many of you have used generative AI for coding?
- Which programming language have you used with generative AI?
- Do you use version control while writing code?

Major GenAI Coding Tools

- GitHub Copilot
  - GPT-4o/o1/o1-mini and Anthropic's Claude-3.5 Sonnet
  - Most popular and well-integrated
- Amazon CodeWhisperer
  - Amazon's proprietary LLMs
  - Strong in AWS-related code
- Cursor
  - GPT-4, Claude-3.5 Sonnet, and Code-LLaMA
  - Specialized code editing features
- Gemini Code Assist
  - Gemini Ultra 1.0
  - Advanced semantic code understanding
  - Integrated with Google Cloud and Android Studio

NCRA • TIFR

Leading Code-Specialized LLMs

- GPT-4 and o1 class models
    - Very good overall code understanding and generation
    - Excellent context handling up to 128K tokens
    - Strong multi-language support
- Claude-3.5 Sonnet
    - Superior code explanation capabilities
    - Strong focus on code safety and best practices
    - Excellent documentation generation
- Code-LLaMA
    - Open source, can run locally
    - Optimized for programming tasks
    - Lower latency than cloud-based models
- Qwen Coder
    - Fine-tuned specifically for code generation
    - Strong performance in Python and JavaScript
    - Can run on consumer hardware

LLM Features by Tool

- Real-time completion:
  - Copilot: Most responsive, context-aware
  - CodeWhisperer: Good for AWS services
  - Cursor: Fast local processing with Code-LLaMA
- Chat capabilities:
  - Copilot Chat: Full code discussions
  - Cursor Chat: Built-in code explanation
  - Gemini Code Assist: Multi-turn code conversations
- Multi-file understanding:
  - Copilot: Best at project context
  - Cursor: Good at refactoring across files
  - Gemini: Strong semantic understanding

NCRA • TIFR

IDE Integration of LLMs

- Visual Studio Code (VSCode):
  - GitHub Copilot (GPT-4, Claude-3)
  - Amazon CodeWhisperer
  - Cursor's VS Code extension
- JetBrains IDEs (PyCharm, IntelliJ, etc.):
  - GitHub Copilot
  - Gemini Code Assist
  - AWS Toolkit with CodeWhisperer
- Android Studio:
  - Gemini Code Assist
  - GitHub Copilot
- Eclipse:
  - CodeWhisperer via AWS Toolkit
  - Tabnine Eclipse plugin

A very common astronomy use case

- You have a set of data points (catalogs, images, spectra, etc.)
- Each datapoint is associated with an error

A very common astronomy use case

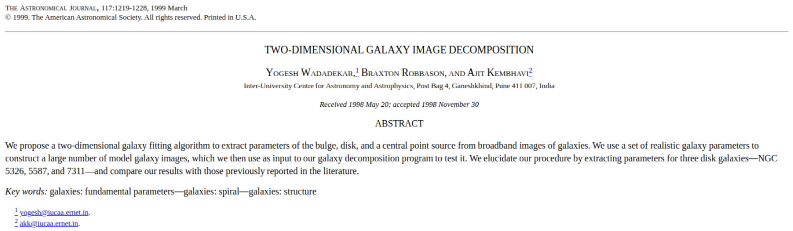
- You have a set of data points (catalogs, images, spectra, etc.)
- Each datapoint is associated with an error
- You have a physically motivated model that you want to fit to the data.
- The model has some free parameters

A very common astronomy use case

- You have a set of data points (catalogs, images, spectra, etc.)
- Each datapoint is associated with an error
- You have a physically motivated model that you want to fit to the data.
- The model has some free parameters
- You want to use a standard optimization algorithm to fit the model to the data
- Most commonly you want to minimize the χ^2 of the model, with each datapoint weighted by its error.
- The best-fit parameters of the model are the ones that minimize the χ^2 and are of scientific interest.

NCRA • TIFR

My first paper



THE ASTRONOMICAL JOURNAL, 117:1219-1228, 1999 March

TWO-DIMENSIONAL GALAXY IMAGE DECOMPOSITION

YOGESH WADADEKAR,[1] BRAXTON ROBBASON, AND AJIT KEMBHAVI[2]
Inter-University Centre for Astronomy and Astrophysics, Post Bag 4, Ganeshkhind, Pune 411 007, India

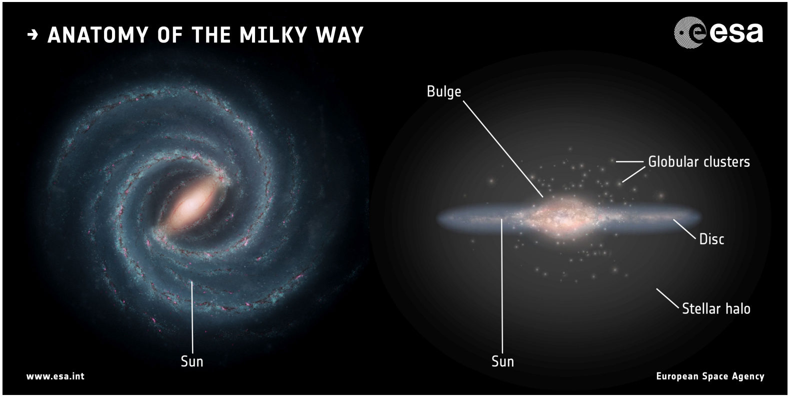
Received 1998 May 20; accepted 1998 November 30

ABSTRACT

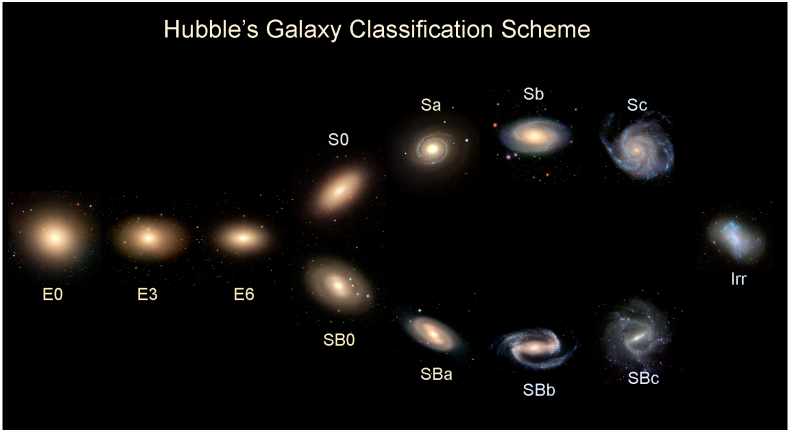
We propose a two-dimensional galaxy fitting algorithm to extract parameters of the bulge, disk, and a central point source from broadband images of galaxies. We use a set of realistic galaxy parameters to construct a large number of model galaxy images, which we then use as input to our galaxy decomposition program to test it. We elucidate our procedure by extracting parameters for three disk galaxies—NGC 5326, 5587, and 7311—and compare our results with those previously reported in the literature.

Key words: galaxies: fundamental parameters—galaxies: spiral—galaxies: structure

[1] yogesh@iucaa.ernet.in.
[2] akk@iucaa.ernet.in.


→ ANATOMY OF THE MILKY WAY
esa
Bulge
Globular clusters
Disc
Stellar halo
Sun
Sun
www.esa.int
European Space Agency

Relative strength of bulge and disk components



The galaxy bulge-disk decomposition problem

As an example, let us take the problem of fitting a galaxy image with a bulge-disk model.

- Obtain a galaxy image from an astronomical survey or your own data.
- Define a model (an analytic function) consisting of a bulge component, a disk component and the sky background.
- Assign initial guesses for the parameters of the components.
- Use an optimization algorithm to fit the model to the galaxy image.
- Minimize the χ^2 value to find the best-fit parameters.
- Analyze the residuals to assess the quality of the fit.
- Given that there will be a number of free parameters in a multidimensional space, you will always need to iterate many times to find the optimal best fit parameters.

NCRA • TIFR

The bulge component modeled by the Sérsic function

- The bulge component of a galaxy is typically modeled using the Sérsic function.
- The Sérsic function is given by:

$$I(r) = I_e \exp\left\{-b_n\left[\left(\frac{r}{r_e}\right)^{1/n} - 1\right]\right\}$$

where $I(r)$ is the intensity at radius r, I_e is the intensity at the effective radius r_e, and n is the Sérsic index.

- The parameter b_n is a function of n and ensures that r_e is the radius enclosing half of the total light.
- The Sérsic index n determines the concentration of the bulge; $n = 4$ corresponds to the de Vaucouleurs profile.

The disk component modeled by the exponential profile

- The disk component of a galaxy is typically modeled using an exponential profile.
- The exponential profile is given by:

$$I(r) = I_0 \exp\left(-\frac{r}{r_d}\right)$$

  where $I(r)$ is the intensity at radius r, I_0 is the central intensity, and r_d is the disk scale length.
- This is actually a special case of the Sérsic profile with $n = 1$.
- The exponential profile is a good description of galaxy disks because stars in the disk follow nearly circular orbits with a density that decreases exponentially with radius.

NCRA • TIFR

Ellipticity and Position Angle

- Both bulge and disk components can be elliptical rather than circular
- The ellipticity ϵ is defined as:

$$\epsilon = 1 - \frac{b}{a}$$

  where *a* and *b* are the semi-major and semi-minor axes
- The position angle (PA) measures the orientation of the major axis
- PA is measured counter-clockwise from North
- These parameters modify how we calculate *r* in the Sérsic and exponential profiles

Additional Free Parameters

- For each component (bulge and disk), we add:
    - Ellipticity (ϵ)
    - Position angle (PA)
    - Center coordinates (x_c, y_c)
- The radius r becomes:

$$r = \sqrt{\left(\frac{x'}{a}\right)^2 + \left(\frac{y'}{b}\right)^2}$$

  where (x', y') are coordinates in the rotated frame
- This adds up to 6 additional parameters per component

The background component

- The background in astronomical images is typically modeled as a constant value.
- This assumes that the sky brightness is uniform across the image.
- The background level can be written simply as:

$$I_{bg} = constant$$

- In reality, the background might have gradients or patterns.
- For more complex cases, polynomial functions can model varying backgrounds.
- Accurate background estimation is crucial for reliable bulge-disk decomposition.

Weighting the datapoints

- Each pixel in the image is a datapoint that contributes to the χ^2
- For Poisson noise, the variance at each pixel is equal to the signal:

$$\sigma_i^2 = S_i + \sigma_{bg}^2$$

where S_i is the signal and σ_{bg}^2 is the background variance

Weighting the datapoints

- Each pixel in the image is a datapoint that contributes to the χ^2
- For Poisson noise, the variance at each pixel is equal to the signal:

$$\sigma_i^2 = S_i + \sigma_{bg}^2$$

  where S_i is the signal and σ_{bg}^2 is the background variance
- The contribution to χ^2 from each pixel is:

$$\chi_i^2 = \frac{(O_i - M_i)^2}{\sigma_i^2}$$

  where O_i is the observed value and M_i is the model value

NCRA • TIFR

Weighting the datapoints

- Each pixel in the image is a datapoint that contributes to the χ^2
- For Poisson noise, the variance at each pixel is equal to the signal:

$$\sigma_i^2 = S_i + \sigma_{bg}^2$$

  where S_i is the signal and σ_{bg}^2 is the background variance
- The contribution to χ^2 from each pixel is:

$$\chi_i^2 = \frac{(O_i - M_i)^2}{\sigma_i^2}$$

  where O_i is the observed value and M_i is the model value
- Alternatively, if a noise map is available:

$$\chi_i^2 = \frac{(O_i - M_i)^2}{N_i^2}$$

  where N_i is the noise value at pixel i

Convolution with the PSF

- The model image must be convolved with the Point Spread Function (PSF) before comparing with data
- The PSF describes how a point source is spread out by the telescope and atmosphere
- Mathematically, the convolution is:

$$I_{conv}(x, y) = \int \int I(x', y') \mathrm{PSF}(x - x', y - y') dx' dy'$$

- The PSF can be measured from stars in the field

NCRA • TIFR

The MINUIT Optimization Engine

- MINUIT was originally developed at CERN by Fred James
- It is widely used for optimization in particle physics
- Key strengths:
    - Robust handling of correlations between parameters
    - Accurate error estimation on fitted parameters
    - Ability to handle both convex (single global minimum, smooth optimization) and non-convex (multiple local minima, more challenging) problems
    - Extensive experience in physics applications

NCRA • TIFR

iminuit: Python Interface to MINUIT

- iminuit is a Python frontend to MINUIT
- Provides the MIGRAD minimizer algorithm
- Features:
    - Fast and accurate minimization using the MIGRAD algorithm
    - MIGRAD uses the Davidon-Fletcher-Powell (DFP) variable metric method
    - Efficient error estimation via the Hesse matrix
    - Support for fixed and limited range parameters
    - Automatic computation of parameter uncertainties
    - Proper handling of parameter limits/bounds
    - Integration with modern Python tools (numpy, scipy)
    - Interactive visualization of fit results
- Can minimize any callable Python function

NCRA • TIFR

Using MIGRAD for Our Problem

- MIGRAD uses variable metric method (quasi-Newton method)
- Approximates the shape of function near minimum as a quadratic form
- Particularly good at handling parameters with different scales
- For our galaxy fitting:
  - Define objective function returning χ^2
  - Set initial parameter values and bounds
  - Call MIGRAD to minimize χ^2 and find best-fit parameters
  - MIGRAD finds the minimum and parameter uncertainties

The fitting process

Video

NCRA • TIFR

Getting Started with GenAI Coding

- GitHub Copilot is one of the best options for getting started with AI-assisted coding
- It integrates seamlessly with Visual Studio Code (VSCode)
- Provides real-time code suggestions as you type
- Understands context from your open files and comments
- Can explain code, suggest improvements, and help with debugging
- Works with multiple programming languages including Python, C++, Julia

Free Access for Students and Teachers

- GitHub offers free access to Copilot through GitHub Education
- Students can get the GitHub Student Developer Pack:
    - Visit education.github.com
    - Sign up with your academic email
    - Verify your student status

Free Access for Students and Teachers

- GitHub offers free access to Copilot through GitHub Education
- Students can get the GitHub Student Developer Pack:
  - Visit education.github.com
  - Sign up with your academic email
  - Verify your student status
- Teachers can apply for GitHub Teacher toolbox:
  - Also at education.github.com
  - Requires proof of faculty status
  - Includes Copilot and other developer tools

NCRA • TIFR

GitHub Copilot's Free Tier (New!)

- As of 18 December 2024, GitHub introduced a free tier for Copilot
- Available to all GitHub users without subscription
- Includes:
  - 50 chat requests in Copilot Chat per month
  - 2000 code completions per month
  - Models available are Claude 3.5 Sonnet and GPT-4o
- Great way to try out AI-assisted coding
- Upgrade to full version available

NCRA • TIFR

GitHub Copilot’s Multi-file Capabilities

- Copilot Chat Edits can help with changes across multiple files
- Copilot Chat Edits can:
  - Make consistent changes across related files
  - Refactor code that spans multiple modules
  - Update function calls in all dependent files
- Simply explain what changes you need in natural language
- Copilot will generate a complete set of changes
- Review and accept/reject changes before they are applied

GitHub Workspace: A Collaborative Environment

- GitHub Workspace provides an integrated development environment
- Combines code editor, version control, and AI assistance
- Enables seamless collaboration between you and Copilot
- Access to project files, documentation, and version history
- Real-time code suggestions and completions

Brainstorming with Github Workspace

- Start by describing your problem in natural language
- Github Workspace helps to:
  - Break down complex problems
  - Suggest potential approaches
  - Identify required libraries and tools
  - Point out potential challenges
- Interactive discussion to refine ideas

NCRA • TIFR

Github Workspace: Inside the Technology

- Github Next's Workspace is an experimental development environment
- Key features:
  - DORA metrics for performance monitoring
  - AI-powered pair programming with code review features
  - Code search across your repositories and public code
  - Context-aware code understanding using embeddings
  - Integration with GitHub Actions and CI/CD pipelines
- Uses a graph-based code analysis engine
- Maintains semantic understanding across sessions
- Provides customizable code templates and snippets

NCRA • TIFR

Implementation Phase

- Write code with real-time AI assistance
- Github Workspace suggests complete functions and blocks
- Get help with:
  - Algorithm implementation
  - Error handling
  - Documentation
  - Unit tests
- Review and verify generated code

NCRA • TIFR

Code Building and Repair in GitHub Workspace

- Intelligent code repair suggestions:
  - Identifies potential bugs and security issues
  - Suggests fixes with explanations
  - Shows impact of changes
- Code building features:
  - Smart auto-completion
  - Function signature hints
  - Type inference
  - Refactoring suggestions
- Real-time syntax checking and linting

Issues and Pull Requests in GitHub Workspace

- AI-assisted issue management:
  - Automatically categorizes issues
  - Suggests relevant assignees
  - Identifies duplicate issues
- Enhanced pull request workflow:
  - Automated code review comments
  - Conflict detection and resolution help
  - Test coverage analysis
  - Documentation update suggestions
- Integration with CI/CD pipelines

The Future of GenAI: Software Agents

- Software agents powered by GenAI will automate complex tasks
- Integration with various development tools and platforms
- Enhanced collaboration between humans and AI
- Increased personalization of coding assistance
- Continuous learning and adaptation to user preferences

Manifestations of GenAI Software Agents in the Near Future

- Advanced code generation and optimization
- Intelligent debugging and error handling
- Seamless multi-language support and interoperability
- Automated project management and deployment
- Enhanced security features through AI-driven analysis

NCRA • TIFR

What I noticed

This is based on my current experience. As LLMs are changing rapidly, some or all of these pointers may not remain valid in the future.

- new codes based on publicly available codes and algorithms are well written by AI. When coding using commonly used languages, libraries and use cases, GenAI performance is very good.
- since most astronomy codes are open source, generating new good quality code with LLMs is easier.
- code generation slows down as the code gets longer.
- GenAI works better when writing codes from scratch than when modifying existing codes
- Stick to one LLM model for your entire code. Changing horses midway is not a good idea.
- AI generated codes are still easy to spot - good quality comments, good variable names etc.

Things to remember

- LLMs usually do not make syntactical errors, but mistakes in logic do occur.
- **Just like when you wrote the code yourself, the responsibility for the correctness of the code is yours alone.**

Improvement in productivity

- Perhaps an order of magnitude improvement in productivity for an expert.
- Zero or even negative impact for a beginner. Do not use them until you develop expertise either in programming or domain, preferably both.
- We need to understand the devastating impact of this on the peer and apprentice model of learning.