Energy Calculations for PTAC vs PTHP Systems

EDK Energy Insight Platform

September 9, 2025

Contents

1 Introduction

This document outlines the energy calculations for comparing Packaged Terminal Air Conditioner (PTAC) and Packaged Terminal Heat Pump (PTHP) systems, using the standardized variable names from our codebase and the AI-determined unit distributions described in the following section.

2 AI-Powered Unit Distribution Analysis

Before calculating energy impacts of PTAC to PTHP conversions, we must first determine the unit distribution within each building. Many NYC buildings lack detailed unit mix data, making AI analysis essential for accurate energy modeling.

2.1 Overview and Purpose

The EDK platform uses artificial intelligence to analyze building characteristics and determine unit distribution when detailed unit mix data is unavailable. This AI-driven approach processes publicly available building data from NYC agencies to infer the most likely unit breakdown for residential buildings.

2.2 Input Data Sources

The AI analysis utilizes two primary data sources:

2.2.1 PLUTO Data (Primary Input)

Data from the NYC Department of City Planning's Primary Land Use Tax Lot Output (PLUTO) dataset:

- `bldgclass`: Building Class Code (e.g., C1, D3, R1, etc.)
- `yearbuilt`: Year the building was constructed
- `unitsres`: Total number of residential units
- `resarea`: Total residential floor area in square feet
- `numfloors`: Number of floors in the building
- `boro`: Borough location (Manhattan, Brooklyn, Queens, Bronx, Staten Island)

2.2.2 Local Law 84 Data (Secondary Input)

Energy consumption data when available:

- `site_eui`: Site Energy Use Intensity
- Energy usage patterns and building performance metrics

2.3 AI Analysis Architecture

The system employs a LangChain-based graph architecture with multiple processing nodes:

1. **Start Node**: Validates input PLUTO data
2. **Inference Node**: Uses OpenAI GPT-4 to analyze building characteristics
3. **Calculation Node**: Converts unit breakdown to PTAC unit count
4. **Validation Node**: Ensures output integrity and constraints

2.4 Prompt Engineering and Structured Output

The AI uses carefully crafted prompts with structured output validation to ensure consistent, accurate results:

```
const prompt = `Analyze a building in ${plutoData.boro} with the following
characteristics and provide a detailed breakdown of residential units:

Building Details:
- Building Class: ${plutoData.bldgclass}
- Year Built: ${plutoData.yearbuilt}
- Total Residential Units: ${plutoData.unitsres}
- Total Residential Area: ${plutoData.resarea} sq ft
- Number of Floors: ${plutoData.numfloors}

Requirements:
1. The sum of all unit types MUST equal exactly ${plutoData.unitsres} total units
2. Consider typical unit layouts and sizes for this building's era and class
3. Account for the total residential area when determining unit mix
4. Consider historical building practices from the construction period

You must return:
1. studio: Number of studio units (must be a non-negative integer)
2. one_bed: Number of one-bedroom units (must be a non-negative integer)
3. two_bed: Number of two-bedroom units (must be a non-negative integer)
4. three_plus: Number of three or more bedroom units (must be a non-negative integer)
5. reasoning: A detailed explanation of your calculation process`;
```

2.5 Structured Output Schema

The AI response follows a strict TypeScript schema to ensure data integrity:

```
interface UnitBreakdown {
  studio: number;      // Number of studio units
  one_bed: number;     // Number of one-bedroom units
  two_bed: number;     // Number of two-bedroom units
  three_plus: number;  // Number of three or more bedroom units
  source: 'AI-Assumed';
}

const unitBreakdownSchema = z.object({
  studio: z.number().int().min(0).describe('Number of studio units'),
  one_bed: z.number().int().min(0).describe('Number of one-bedroom units'),
  two_bed: z.number().int().min(0).describe('Number of two-bedroom units'),
  three_plus: z.number().int().min(0).describe('Number of three or more bedroom units'),
  reasoning: z.string().describe('Detailed explanation of the unit breakdown calculation')
});
```

2.6 Unit Mix to PTAC Unit Calculation

Once the AI determines the unit breakdown, the system calculates the total number of PTAC units using industry-standard assumptions about air conditioning needs per room:

2.6.1 PTAC Unit Allocation Rules

- **Studio Units**: 1 PTAC unit (combined living/sleeping area)
- **One-Bedroom Units**: 2 PTAC units (1 bedroom + 1 living area)
- **Two-Bedroom Units**: 3 PTAC units (2 bedrooms + 1 living area)
- **Three+ Bedroom Units**: 4 PTAC units (3 bedrooms + 1 living area)

2.6.2 PTAC Calculation Formula

The total building PTAC unit count is calculated as:

$$N_{\text{ptacUnits}} = N_{\text{studio}} \times 1 + N_{\text{oneBed}} \times 2 + N_{\text{twoBed}} \times 3 + N_{\text{threePlus}} \times 4 \tag{1}$$

Where:

- N_{studio}: Number of studio units
- N_{oneBed}: Number of one-bedroom units
- N_{twoBed}: Number of two-bedroom units
- $N_{\text{threePlus}}$: Number of three or more bedroom units

2.7 Implementation Example

For a 1960s 6-story building with 24 residential units and 18,000 sq ft of residential area:

```
// AI Analysis Result
{
  "studio": 6,
  "one_bed": 12,
  "two_bed": 6,
  "three_plus": 0,
  "reasoning": "Based on the building's 1960s construction era and 750 sq ft average unit size,
      this represents a typical mid-century apartment building layout with a mix of studios and
      one-bedrooms, plus some larger two-bedroom units for families."
}

// PTAC Unit Calculation
ptacUnits = 6×1 + 12×2 + 6×3 + 0×4 = 6 + 24 + 18 + 0 = 48 total PTAC units
```

2.8 Validation and Quality Control

The system implements multiple validation layers:

1. **Sum Validation**: Unit breakdown must equal PLUTO total residential units
2. **Non-Zero Check**: Prevents all-zero responses that would indicate analysis failure
3. **Integer Validation**: Ensures all unit counts are non-negative integers
4. **Reasoning Requirement**: AI must provide detailed explanation for transparency

If validation fails, the system throws an error and requests re-analysis rather than proceeding with invalid data.

2.9 AI Analysis Output

The complete AI analysis provides:

- Detailed unit breakdown with counts for each unit type
- AI reasoning explaining the analysis logic
- Total PTAC unit count for energy calculations
- Data source attribution (`source: 'AI-Assumed'`)
- Validation confirmation of constraint compliance

This AI-generated unit distribution then feeds into all subsequent energy calculations, cost analyses, and LL97 compliance assessments covered in the following sections.

3 PTAC System Calculations

3.1 Per-Unit Constants

The system uses fundamental constants for PTAC units in original units and their MMBtu equivalents:

- `annualUnitThermsHeatingPTAC`: 255 therms per year per unit for heating
- `annualUnitKwhCoolingPTAC`: 16,000 kWh per year per unit for cooling
- `annualUnitMMBtuHeatingPTAC`: 25.5 MMBtu per year per unit for heating
  - Converted from 255 therms $\times$ 0.1 MMBtu/therm = 25.5 MMBtu
- `annualUnitMMBtuCoolingPTAC`: 5.459427 MMBtu per year per unit for cooling
  - Converted from 16,000 kWh $\times$ 0.003412 MMBtu/kWh = 5.459427 MMBtu

3.2 Building-Level Calculations

For a building with multiple units, we calculate both MMBtu and original unit values:

3.2.1 MMBtu Building Totals

$$\texttt{annualBuildingMMBtuCoolingPTAC} = N_{ptacUnits} \times \texttt{annualUnitMMBtuCoolingPTAC} \tag{2}$$

$$\texttt{annualBuildingMMBtuHeatingPTAC} = N_{ptacUnits} \times \texttt{annualUnitMMBtuHeatingPTAC} \tag{3}$$

$$\texttt{annualBuildingMMBtuTotalPTAC} = \texttt{annualBuildingMMBtuCoolingPTAC} + \texttt{annualBuildingMMBtuHeatingPTAC} \tag{4}$$

4 PTHP System Calculations

4.1 Heating Energy Conversion with Building-Specific EFLH

For PTHP systems, the heating energy is calculated using building-specific Equivalent Full Load Hours (EFLH) derived from PLUTO data, combined with PTHP heating capacity and coefficient of performance:

4.1.1 Step 1: Calculate Annual Building kWh for PTHP Heating

The annual building heating consumption in kWh is calculated using the detailed equation:

$$\texttt{annualBuildingkWhHeatingPTHP} = \frac{\texttt{heatingCapacityPTHP}}{3.412} \times \frac{1}{\texttt{pthpCOP}} \times \text{EFLH} \times N_{ptacUnits} \tag{5}$$

Where:

- `heatingCapacityPTHP`: 8 KBtu (constant heating capacity per PTHP unit)
- 3.412: Conversion factor from KBtu to kW (kW per KBtu)
- `pthpCOP`: 1.51 (Coefficient of Performance for PTHP)
- EFLH: Equivalent Full Load Hours (building-specific, from PLUTO data)
- $N_{ptacUnits}$: Number of PTAC units to be replaced

4.1.2 Step 2: EFLH Lookup from PLUTO Data

The EFLH value is determined by building characteristics from PLUTO data using the following function:

```
export function getEFLHFromPluto(yearBuilt: number, floors: number): number {
  const buildingType = floors <= 6 ? 'lowRise' : 'highRise';

  const constructionEra = yearBuilt <= 1939 ? 'prewar' :
                         yearBuilt <= 1978 ? 'pre79' :
                         yearBuilt <= 2006 ? 'post1979' : 'post2007';

  const eflhTable = {
    lowRise: { prewar: 974, pre79: 738, post1979: 705, post2007: 491 },
    highRise: { prewar: 987, pre79: 513, post1979: 385, post2007: 214 }
  };

  return eflhTable[buildingType][constructionEra];
}
```

The EFLH table categorizes buildings by:

- **Building Type**: Low-rise ($\leq$ 6 floors) vs High-rise ($>$ 6 floors)
- **Construction Era**: Pre-war ($\leq$ 1939), Pre-79 (1940-1978), 1979-2006, 2007-Present
- **EFLH Range**: 214-987 hours depending on building characteristics

4.1.3 Step 3: Convert kWh to MMBtu

Finally, convert the kWh result to MMBtu for consistency with other calculations:

$$\texttt{annualBuildingMMBtuHeatingPTHP} = \texttt{annualBuildingkWhHeatingPTHP} \times 0.003412 \tag{6}$$

Where 0.003412 is the conversion factor from kWh to MMBtu.

4.2 Cooling Energy

The cooling energy for PTHP remains equivalent to PTAC:

$$\texttt{annualBuildingMMBtuCoolingPTHP} = \texttt{annualBuildingMMBtuCoolingPTAC} \tag{7}$$

4.3 Total PTHP Energy

The total annual building energy consumption for PTHP:

$$\texttt{annualBuildingMMBtuTotalPTHP} = \texttt{annualBuildingMMBtuCoolingPTHP} + \texttt{annualBuildingMMBtuHeatingPTHP} \tag{8}$$

5 Energy Reduction Analysis

The energy reduction achieved by switching from PTAC to PTHP is calculated as:

$$\text{Reduction (\%)} = \frac{\texttt{annualBuildingMMBtuTotalPTAC} - \texttt{annualBuildingMMBtuTotalPTHP}}{\texttt{annualBuildingMMBtuTotalPTAC}} \times 100\% \tag{9}$$

6 Total Retrofit Cost Calculation

The total cost for retrofitting from PTAC to PTHP systems includes unit costs, installation costs, and contingency:

$$\texttt{totalRetrofitCost} = (\texttt{pthpUnitCost} + \texttt{pthpInstallationCost}) \times \text{Total PTHP Units} \times (1 + \texttt{pthpContingency}) \tag{10}$$

Example calculation:

$$\texttt{totalRetrofitCost} = (\$1,100 + \$450) \times N_{ptacUnits} \times 1.10 = \$1,705 \times N_{ptacUnits} \tag{11}$$

Where:

- `pthpUnitCost`: $1,100 per PTHP unit
- `pthpInstallationCost`: $450 per unit
- `pthpContingency`: 10% (1.10 multiplier)
- $N_{ptacUnits}$: Total number of PTHP units to be installed

7 Energy Cost Savings Calculation

To calculate costs, we need building totals in kWh and therms.

7.1 PTAC Building Totals for Cost Calculations

$$\texttt{annualBuildingThermsHeatingPTAC} = N_{ptacUnits} \times \texttt{annualUnitThermsHeatingPTAC} \tag{12}$$

$$\texttt{annualBuildingKwhCoolingPTAC} = N_{ptacUnits} \times \texttt{annualUnitKwhCoolingPTAC} \tag{13}$$

Where `annualUnitKwhCoolingPTAC` = 16,000 kWh per unit per year.

7.2 PTHP Building Totals for Cost Calculations

For PTHP systems, the kWh values needed for cost calculations are:

- `annualBuildingkWhHeatingPTHP`: Already calculated in Section 4.1.1 using EFLH methodology
- `annualBuildingKwhCoolingPTHP`: Same as PTAC cooling (PTHP cooling efficiency assumed equivalent)

$$\texttt{annualBuildingKwhCoolingPTHP} = \texttt{annualBuildingKwhCoolingPTAC} = N_{ptacUnits} \times 16,000 \tag{14}$$

7.3 Cost Calculations

The annual energy cost savings from switching to PTHP systems is calculated as:

$$\begin{aligned}\texttt{annualBuildingCostPTAC} = {} & \texttt{annualBuildingKwhCoolingPTAC} \times \texttt{priceKwhHour} \\ & + \texttt{annualBuildingThermsHeatingPTAC} \times \texttt{priceThermHour}\end{aligned} \tag{15}$$

$$\begin{aligned}\texttt{annualBuildingCostPTHP} = {} & \texttt{annualBuildingKwhHeatingPTHP} \times \texttt{priceKwhHour} \\ & + \texttt{annualBuildingKwhCoolingPTHP} \times \texttt{priceKwhHour}\end{aligned} \tag{16}$$

$$\texttt{annualSavingsEnergy} = \texttt{annualBuildingCostPTAC} - \texttt{annualBuildingCostPTHP} \tag{17}$$

Where:

- `priceKwhHour`: \$0.24 per kilowatt-hour for electricity (NYC average)
- `priceThermHour`: \$1.45 per therm for natural gas (NYC average)[1]

8 LL97 Emissions Savings and Annual Fee Calculation

8.1 Current Building Emissions and Budget

The building's current emissions are obtained from LL84 reporting, specifically from the `total_location_based_ghg` field in the NYC Local Law 84 dataset[2]:

$$\texttt{totalBuildingEmissionsLL84} = \text{LL84[total\_location\_based\_ghg]} \tag{18}$$

The property use breakdown is obtained from the `list_of_all_property_use` field, which contains a comma-separated string of property types with their corresponding square footage. For example: "Office (264550.0), Retail Store (21700.0), Parking (13000.0)". This string is parsed to extract individual property use types and their associated square footage for emissions budget calculations.

The emissions budget for the building is calculated as the sum of type-specific emissions limits applied to corresponding square footage[3]:

$$\texttt{emissionsBudget} = \sum_i \texttt{squareFootageByType}_i \times \texttt{emissionsLimit}_i \tag{19}$$

8.2 Annual Fee Calculation with BE Credit Integration

LL97 has four compliance periods with different emissions budgets. For buildings implementing PTHP upgrades, the annual fee calculation includes Beneficial Electrification (BE) credits that reduce penalties. The calculation process follows these steps for each period:

1. Calculate base emissions budget for the period 2. Determine adjusted building emissions after retrofit 3. Calculate BE credits (if applicable) 4. Apply BE credits to reduce adjusted emissions 5. Calculate final annual fee based on credited emissions

8.2.1 Base Emissions Budgets by Period

2024-2029 Compliance Period

$$\texttt{emissionsBudget2024to2029} = \sum_i \texttt{squareFootageByType}_i \times \texttt{emissionsLimit2024to2029}_i \tag{20}$$

[1]Heating oil is priced at \$2.60-\$2.77 per therm, but most buildings use natural gas.

[2]NYC Open Data API: https://data.cityofnewyork.us/resource/5zyy-y8am.json using `total_location_based_ghg` field. This field contains the total GHG emissions in metric tons CO2e calculated using location-based emissions factors.

[3]Key LL97 property type limits (tCO2e/sf) across four compliance periods: Office (2024-29: 0.00846, 2030-34: 0.00453, 2035-39: 0.00165234, 2040-49: 0.000581893), Multifamily Housing (2024-29: 0.00892, 2030-34: 0.00453), Hospital (2024-29: 0.02551, 2030-34: 0.01542). Complete mapping available in ll97_espm_to_bc_caps_with_2035_2049.json.

2030-2034 Compliance Period

$$\texttt{emissionsBudget2030to2034} = \sum_i \texttt{squareFootageByType}_i \times \texttt{emissionsLimit2030to2034}_i \tag{21}$$

2035-2039 Compliance Period

$$\texttt{emissionsBudget2035to2039} = \sum_i \texttt{squareFootageByType}_i \times \texttt{emissionsLimit2035to2039}_i \tag{22}$$

2040-2049 Compliance Period

$$\texttt{emissionsBudget2040to2049} = \sum_i \texttt{squareFootageByType}_i \times \texttt{emissionsLimit2040to2049}_i \tag{23}$$

Where `feePerTonCO2e` = \$268 per tCO2e over budget, and each subsequent period has more stringent (lower) emissions limits.

8.2.2 Adjusted Building Emissions After Retrofit

To accurately assess the LL97 fee impact of the PTAC→PTHP retrofit, we must calculate the building's adjusted emissions that reflect the actual post-retrofit energy consumption and emissions profile, rather than using the baseline LL84 emissions[4].

The adjusted building emissions are calculated as:

$$\begin{aligned}\texttt{adjustedTotalBuildingEmissions} = {} & \texttt{totalBuildingEmissionsLL84} \\ & - (\texttt{annualBuildingMMBtuHeatingPTAC} \times \texttt{efGas}) \\ & + (\texttt{annualBuildingKwhHeatingPTHP} \times \texttt{efGrid\_kWh(year)})\end{aligned} \tag{24}$$

Where the emissions factors are:

- `efGas`: 0.05311 tCO e/MMBtu (natural gas factor, constant 2024-2034)
- `efGrid_kWh(year)`: Grid electricity factor (tCO e/kWh) varying by compliance period:
  - `efGrid2024to2029`: 0.000288962 tCO e/kWh
  - `efGrid2030to2034`: 0.000145 tCO e/kWh
  - `efGrid2035toFuture`: 0.000145 tCO e/kWh (assumed same as 2030-2034, pending further research[5])

This gives us period-specific adjusted emissions:

$$\begin{aligned}\texttt{adjustedTotalBuildingEmissions2024to2029} = {} & \texttt{totalBuildingEmissionsLL84} \\ & - (\texttt{annualBuildingMMBtuHeatingPTAC} \times 0.05311) \\ & + (\texttt{annualBuildingKwhHeatingPTHP} \times 0.000288962)\end{aligned} \tag{25}$$

$$\begin{aligned}\texttt{adjustedTotalBuildingEmissions2030to2034} = {} & \texttt{totalBuildingEmissionsLL84} \\ & - (\texttt{annualBuildingMMBtuHeatingPTAC} \times 0.05311) \\ & + (\texttt{annualBuildingKwhHeatingPTHP} \times 0.000145)\end{aligned} \tag{26}$$

[4]Using raw LL84 emissions would not account for the actual emissions reduction from switching from gas heating to electric heat pumps, which varies by time period due to changing grid electricity emissions factors.

[5]The grid electricity emissions factor for periods beyond 2034 requires additional research to determine accurate projections based on grid decarbonization plans and renewable energy adoption rates.

$$\begin{aligned}\texttt{adjustedTotalBuildingEmissions2035to2039} = {} & \texttt{totalBuildingEmissionsLL84} \\ & - (\texttt{annualBuildingMMBtuHeatingPTAC} \times 0.05311) \\ & + (\texttt{annualBuildingKwhHeatingPTHP} \times \texttt{efGrid2035toFuture})\end{aligned} \tag{27}$$

$$\begin{aligned}\texttt{adjustedTotalBuildingEmissions2040to2049} = {} & \texttt{totalBuildingEmissionsLL84} \\ & - (\texttt{annualBuildingMMBtuHeatingPTAC} \times 0.05311) \\ & + (\texttt{annualBuildingKwhHeatingPTHP} \times \texttt{efGrid2035toFuture})\end{aligned} \tag{28}$$

8.2.3 Beneficial Electrification (BE) Credit Calculation

The BE credit is calculated based on heating electrification only, using annual heating kWh and time-dependent coefficients:

$$\texttt{beCreditBefore2027} = \texttt{annualBuildingkWhHeatingPTHP} \times 0.0013 \tag{29}$$

$$\texttt{beCredit2027to2029} = \texttt{annualBuildingkWhHeatingPTHP} \times 0.00065 \tag{30}$$

Where:

- Before January 1, 2027: Coefficient = 0.0013 tCO e/kWh
- 2027-2029: Coefficient = 0.00065 tCO e/kWh

8.2.4 Final Annual Fee Calculation with BE Credits

Annual Fee for 2024-2026 (with BE Credit)

For buildings upgrading before January 1, 2027:

$$\texttt{creditedEmissionsBefore2027} = \texttt{adjustedTotalBuildingEmissions2024to2029} - \texttt{beCreditBefore2027} \tag{31}$$

$$\begin{aligned}\texttt{adjustedAnnualFeeBefore2027} = {} & (\texttt{creditedEmissionsBefore2027} \\ & - \texttt{emissionsBudget2024to2029}) \times \texttt{feePerTonCO2e}\end{aligned} \tag{32}$$

Annual Fee for 2027-2029 (with Reduced BE Credit)

For buildings upgrading between 2027-2029:

$$\texttt{creditedEmissions2027to2029} = \texttt{adjustedTotalBuildingEmissions2024to2029} - \texttt{beCredit2027to2029} \tag{33}$$

$$\begin{aligned}\texttt{adjustedAnnualFee2027to2029} = {} & (\texttt{creditedEmissions2027to2029} \\ & - \texttt{emissionsBudget2024to2029}) \times \texttt{feePerTonCO2e}\end{aligned} \tag{34}$$

Annual Fee for 2030-2034 (No BE Credit)

For the 2030-2034 period, there is no BE credit available, so the fee is calculated using the adjusted emissions for this period:

$$\texttt{adjustedAnnualFee2030to2034} = (\texttt{adjustedTotalBuildingEmissions2030to2034} - \texttt{emissionsBudget2030to2034}) \times \texttt{feePerTonCO2e} \quad (35)$$

Annual Fee for 2035-2039 (No BE Credit)

For the 2035-2039 period, there is no BE credit available, so the fee is calculated using the adjusted emissions for this period:

$$\texttt{adjustedAnnualFee2035to2039} = (\texttt{adjustedTotalBuildingEmissions2035to2039} - \texttt{emissionsBudget2035to2039}) \times \texttt{feePerTonCO2e} \quad (36)$$

Annual Fee for 2040-2049 (No BE Credit)

For the 2040-2049 period, there is no BE credit available, so the fee is calculated using the adjusted emissions for this period:

$$\texttt{adjustedAnnualFee2040to2049} = (\texttt{adjustedTotalBuildingEmissions2040to2049} - \texttt{emissionsBudget2040to2049}) \times \texttt{feePerTonCO2e} \quad (37)$$

For both periods, we use the `efGrid2035toFuture` emissions factor as defined above.

8.2.5 Baseline Annual Fees (No Upgrade)

For comparison, the baseline annual fees without any retrofit are calculated using the original LL84 emissions:

$$\texttt{annualFeeExceedingBudget2024to2029} = (\texttt{totalBuildingEmissionsLL84} - \texttt{emissionsBudget2024to2029}) \times \texttt{feePerTonCO2e} \quad (38)$$

$$\texttt{annualFeeExceedingBudget2030to2034} = (\texttt{totalBuildingEmissionsLL84} - \texttt{emissionsBudget2030to2034}) \times \texttt{feePerTonCO2e} \quad (39)$$

$$\texttt{annualFeeExceedingBudget2035to2039} = (\texttt{totalBuildingEmissionsLL84} - \texttt{emissionsBudget2035to2039}) \times \texttt{feePerTonCO2e} \quad (40)$$

$$\texttt{annualFeeExceedingBudget2040to2049} = (\texttt{totalBuildingEmissionsLL84} - \texttt{emissionsBudget2040to2049}) \times \texttt{feePerTonCO2e} \quad (41)$$

9 Financial Analysis of PTHP Upgrade

In order to calculate the financial viability of upgrading from PTAC to PTHP systems, we first need to calculate how much money you will be saving each year from the time you upgrade. This annual savings is composed of two main components: the energy savings (`annualSavingsEnergy` from Section 7) and the LL97 fee avoidance. The following 4 subsections show how calculating this varies depending on which period you are in, noting that LL97 compliance continues post-2035 with ongoing fee avoidance benefits.

9.1 LL97 Fee Avoidance Calculations

9.1.1 LL97 Fee Avoidance for 2024-2027

$$\begin{aligned}\texttt{annualLL97FeeAvoidance2024to2027} &= \texttt{annualFeeExceedingBudget2024to2029} \\ &- \texttt{adjustedAnnualFeeBefore2027}\end{aligned} \tag{42}$$

9.1.2 LL97 Fee Avoidance for 2027-2029

$$\begin{aligned}\texttt{annualLL97FeeAvoidance2027to2029} &= \texttt{annualFeeExceedingBudget2024to2029} \\ &- \texttt{adjustedAnnualFee2027to2029}\end{aligned} \tag{43}$$

9.1.3 LL97 Fee Avoidance for 2030-2034

$$\begin{aligned}\texttt{annualLL97FeeAvoidance2030to2034} &= \texttt{annualFeeExceedingBudget2030to2034} \\ &- \texttt{adjustedAnnualFee2030to2034}\end{aligned} \tag{44}$$

9.1.4 LL97 Fee Avoidance for 2035-2039

$$\begin{aligned}\texttt{annualLL97FeeAvoidance2035to2039} &= \texttt{annualFeeExceedingBudget2035to2039} \\ &- \texttt{adjustedAnnualFee2035to2039}\end{aligned} \tag{45}$$

9.1.5 LL97 Fee Avoidance for 2040-2049

$$\begin{aligned}\texttt{annualLL97FeeAvoidance2040to2049} &= \texttt{annualFeeExceedingBudget2040to2049} \\ &- \texttt{adjustedAnnualFee2040to2049}\end{aligned} \tag{46}$$

9.2 Code Implementation

This is the implementation for calculating cumulative savings:

Code Chunk 1: Function to calculate annual savings in a given year

```
function getAnnualSavings(year: number): number {
  const energySavings = annualSavingsEnergy;

  if (year >= 2024 && year <= 2026) {
    return energySavings + annualLL97FeeAvoidance2024to2027;
  } else if (year >= 2027 && year <= 2029) {
    return energySavings + annualLL97FeeAvoidance2027to2029;
  } else if (year >= 2030 && year <= 2034) {
    return energySavings + annualLL97FeeAvoidance2030to2034;
  } else if (year >= 2035 && year <= 2039) {
    return energySavings + annualLL97FeeAvoidance2035to2039;
  } else {
    return energySavings + annualLL97FeeAvoidance2040to2049;
  }
}
```

Code Chunk 2: Example usage of calculating annual savings

```
// Loan term years for a static example - upgrade completes in 2025, savings start in 2026
const loanYears = [2025, 2026, 2027, 2028, 2029, 2030, 2031, 2032, 2033, 2034, 2035, 2036, 2037,
    2038, 2039, 2040];
const cumulativeSavingsByYear = [];
let cumulativeSavings = 0;

for (const year of loanYears) {
  // No savings in 2025 (upgrade year), savings begin in 2026
  const annualSavings = year === 2025 ? 0 : getAnnualSavings(year);
  cumulativeSavings += annualSavings;

  cumulativeSavingsByYear.push(cumulativeSavings);
}

// This array will be the data for the green cumulative savings line in the visualization
console.log(cumulativeSavingsByYear);

// Actual output: [0, 47000, 94000, 138000, 182000, 226000, 267000, 308000, 349000, 390000,
    431000, 468000, 505000, 542000, 579000, 616000]
```

From the array of cumulative savings year-by-year, once it gets above the totalRetrofitCost, then that year is the Simple Payback Period.[6]

9.3 Payback Period Implementation

For `totalRetrofitCost` = \$500,000, the cumulative savings array shows:

[0, 47000, 94000, 138000, 182000, 226000, 267000, 308000, 349000, 390000, 431000, 468000, **505000**, 542000, 579000, 616000]

Year 2037 achieves payback with \$505,000 cumulative savings.

10 Loan Financing and Payback Visualization

This section demonstrates how loan financing affects the payback period through a visual representation of loan balance versus cumulative savings over time.

[6]Simple division (totalRetrofitCost ÷ averageAnnualSavings) cannot be used here because annual savings vary significantly across LL97 compliance periods. The year-by-year calculation accounts for these varying savings rates.

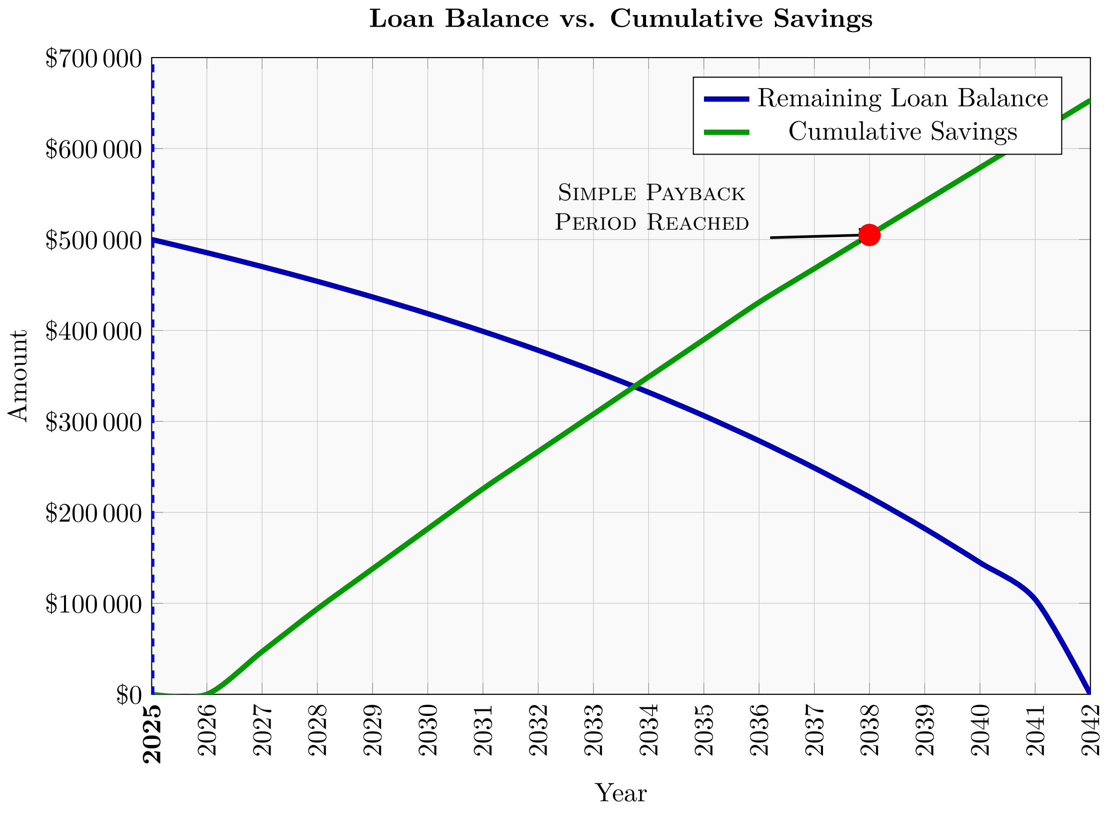
10.1 Loan Parameters

For this example, we use typical commercial loan parameters[7]:

The loan calculations use standard amortization formulas[8] and the remaining balance formula[9].

Cumulative savings grow linearly over time[10]:

10.2 Visualization



The simple payback period occurs when the green cumulative savings line reaches the loan principal amount of $500,000. This visualization shows how cumulative savings continue to grow beyond payback, demonstrating the long-term financial benefits of the retrofit investment.

11 NOI Analysis

Net Operating Income (NOI) represents the annual income generated by a property after deducting all operating expenses but before deducting taxes and debt service. The methodology uses a unified approach based on rent stabilization status and building characteristics from PLUTO data.

11.1 NOI Calculation Methodology

The system uses a three-step process to calculate annual building NOI:

[7]Example parameters: Principal = $500,000 (`totalRetrofitCost`), Term = 15 years, Annual interest rate = 6%, Monthly interest rate = 0.005, Total payments = 180.

[8]Monthly payment: $P \times \frac{r(1+r)^n}{(1+r)^n-1}$ where P=principal, r=monthly rate, n=total payments.

[9]Remaining balance: $P \times \frac{(1+r)^n-(1+r)^{mt}}{(1+r)^n-1}$ where t=years elapsed, m=12 months/year.

[10]Simplified as: Cumulative Savings(t) = $t \times$ Average Annual Savings. In reality, LL97 fee avoidance varies by compliance period.

11.1.1 Step 1: NOI Calculation

The system uses a three-tier approach based on building class:

Cooperative Buildings (Building Classes C0-C9)

NOI values are obtained directly from the NYC Open Data Cooperative Comparable Rental Income dataset[11]:

$$\texttt{currentNOI} = \text{API(BBL)[net\_operating\_income]} \tag{47}$$

Condominium Buildings (Building Classes R4, R5, R6A-R9B, D4-D9)

NOI values are obtained directly from the NYC Open Data Condominium Comparable Rental Income dataset[12]:

$$\texttt{currentNOI} = \text{API(BBL)[net\_operating\_income]} \tag{48}$$

All Other Residential Buildings (Fallback)

For buildings not covered by NYC Open Data APIs or where data is not available, NOI is calculated using the 2025 RGB Study data[13]. To use this data, we need to determine rent stabilization status and location:

11.1.2 Step 2: Rent Stabilization Determination

Determine if the building contains rent stabilized units:

```
function isRentStabilized(bbl: string, pluto: PlutoRecord): boolean {
  // Primary Method: BBL lookup in rent stabilized buildings registry
  if (rentStabilizedBBLs.includes(bbl)) {
    return true;
  }

  // Fallback Heuristic: PLUTO-based criteria
  const isPrewar = pluto.yearBuilt > 0 && pluto.yearBuilt < 1974;
  const isHighRise = pluto.numFloors > 6;
  const isNotCondo = pluto.condoNo == null;
  const isNotCoop = !pluto.bldgClass.includes("C6") && !pluto.bldgClass.includes("D4");

  return isPrewar && isHighRise && isNotCondo && isNotCoop;
}
```

11.1.3 Step 3: Location Assignment

Determine the building's location category based on borough and community district (Core Manhattan: CD 101-108; Upper Manhattan: CD 109-112).

11.1.4 Step 4: Final NOI Calculation

The NOI per unit per month is obtained from the 2025 RGB Study data using location, rent stabilization status, building era (for stabilized buildings), and size buckets (11-19 units, 20-99 units, 100+ units). Era is determined by: $\texttt{yearBuilt} \leq 1973 \rightarrow \texttt{pre\_1974}$, otherwise $\texttt{post\_1973}$. Market rate data is not segmented by building era.

For all buildings, the final annual building NOI is calculated as:

$$\texttt{annualBuildingNOI} = \texttt{noiPerUnitPerMonth} \times \texttt{unitsRes} \times 12 \tag{49}$$

[11]NYC Open Data API: https://data.cityofnewyork.us/resource/myei-c3fa.json using `net_operating_income` field. Data updated periodically by NYC Department of Finance.

[12]NYC Open Data API: https://data.cityofnewyork.us/resource/9ck6-2jew.json using `net_operating_income` field. Data updated periodically by NYC Department of Finance.

[13]Source: *2025 Income and Expense Study*, NYC Rent Guidelines Board, March 2025. Data structured by tenure (stabilized/market rate), location, building age, and size buckets.

Where:

- `noiPerUnitPerMonth`: NOI per residential unit per month from lookup
- `unitsRes`: Total residential units in building
- 12: Months per year conversion factor

11.2 Upgrade Impact on NOI

The true impact of PTHP upgrades becomes clear when comparing NOI scenarios:

Scenario A: No Upgrade (Status Quo)

$$\texttt{noiNoUpgrade2024to2029} = \texttt{currentNOI} - \texttt{annualFeeExceedingBudget2024to2029} \tag{50}$$

$$\texttt{noiNoUpgrade2030to2034} = \texttt{currentNOI} - \texttt{annualFeeExceedingBudget2030to2034} \tag{51}$$

$$\texttt{noiNoUpgradePost2035} = \texttt{currentNOI} - \texttt{annualFeeExceedingBudget2035to2039} \tag{52}$$

Scenario B: With PTHP Upgrade

$$\begin{aligned}\texttt{noiWithUpgrade2024to2027} = \texttt{currentNOI} + \texttt{annualSavingsEnergy}\\ - \texttt{adjustedAnnualFee2024to2027}\end{aligned} \tag{53}$$

$$\begin{aligned}\texttt{noiWithUpgrade2027to2029} = \texttt{currentNOI} + \texttt{annualSavingsEnergy}\\ - \texttt{adjustedAnnualFee2027to2029}\end{aligned} \tag{54}$$

$$\begin{aligned}\texttt{noiWithUpgrade2030to2034} = \texttt{currentNOI} + \texttt{annualSavingsEnergy}\\ - \texttt{adjustedAnnualFee2030to2034}\end{aligned} \tag{55}$$

$$\begin{aligned}\texttt{noiWithUpgradePost2035} = \texttt{currentNOI} + \texttt{annualSavingsEnergy}\\ - \texttt{adjustedAnnualFee2035to2039}\end{aligned} \tag{56}$$

11.3 NOI Calculation Functions

The NOI calculations can be implemented with unified functions that handle all compliance periods:

```
function calculateAdjustedNOINoUpgrade(year: number): number {
  const currentNOI = await getNoiByBbl(bbl, unitBreakdown); // from NOI API endpoint

  if (year >= 2024 && year <= 2029) {
    return currentNOI - annualFeeExceedingBudget2024to2029;
  } else if (year >= 2030 && year <= 2034) {
    return currentNOI - annualFeeExceedingBudget2030to2034;
  } else if (year >= 2035 && year <= 2039) {
    return currentNOI - annualFeeExceedingBudget2035to2039;
  } else {
    return currentNOI - annualFeeExceedingBudget2040to2049;
  }
}

function calculateAdjustedNOIUpgrade(year: number): number {
  const currentNOI = await getNoiByBbl(bbl, unitBreakdown); // from NOI API endpoint
  const energySavings = getEnergySavings();

  let reducedLL97Penalties = 0;
  if (year >= 2024 && year <= 2029) {
    reducedLL97Penalties = adjustedAnnualFee2024to2029; // actual penalties paid after upgrade
  } else if (year >= 2030 && year <= 2034) {
    reducedLL97Penalties = adjustedAnnualFee2030to2034; // actual penalties paid after upgrade
  } else if (year >= 2035 && year <= 2039) {
    reducedLL97Penalties = adjustedAnnualFee2035to2039; // actual penalties paid after upgrade
  } else {
    reducedLL97Penalties = adjustedAnnualFee2040to2049; // actual penalties paid after upgrade
  }

  return currentNOI + energySavings - reducedLL97Penalties;
}
```

11.3.1 Implementation Example

For a 100,000 sq ft rental building with current NOI of $1,200,000:

Without Upgrade: Building faces ongoing LL97 penalties starting in 2026

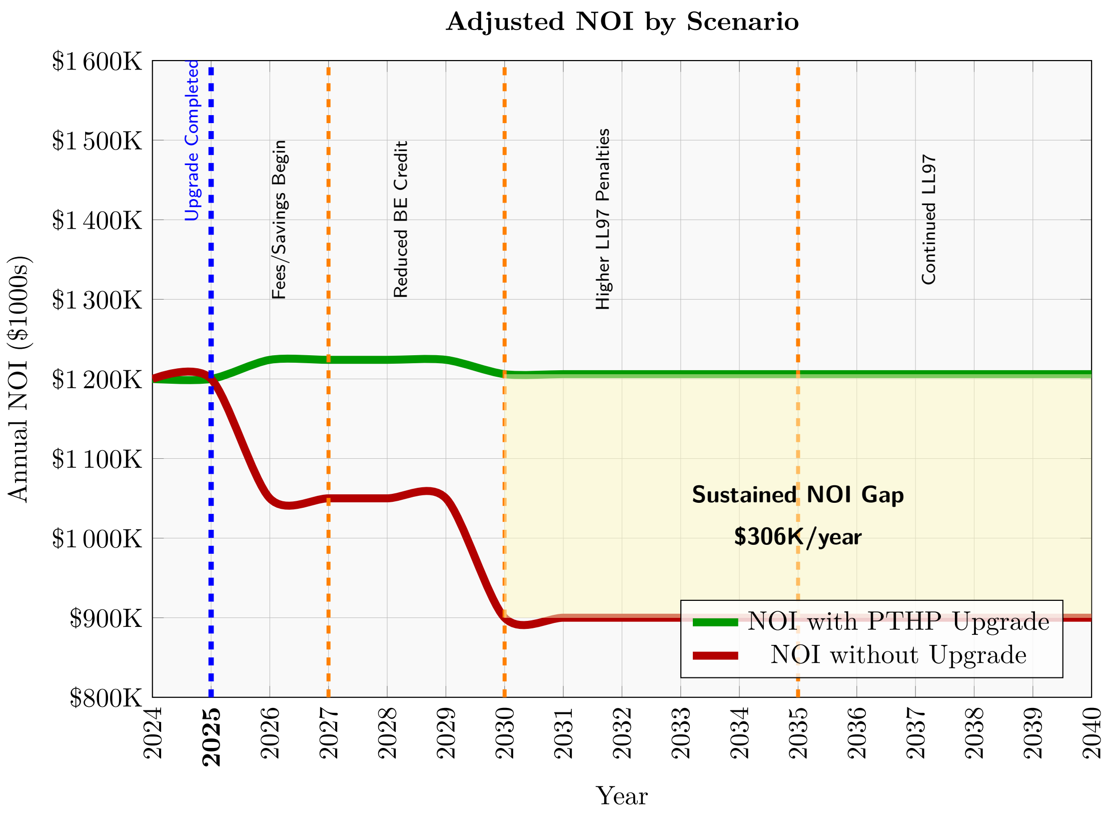
- 2024-2025: NOI = $1,200,000 (current NOI) - $0 (no penalties yet) = $1,200,000
- 2026-2029: NOI = $1,200,000 (current NOI) - $150,000 (LL97 penalties paid) = $1,050,000
- 2030-2034: NOI = $1,200,000 (current NOI) - $300,000 (LL97 penalties paid) = $900,000
- 2035+: NOI = $1,200,000 (current NOI) - $300,000 (LL97 penalties paid) = $900,000

With PTHP Upgrade: Upgrade completes in 2025, savings + reduced penalties start in 2026

- 2024-2025: NOI = $1,200,000 (current NOI) - $0 (no changes yet) = $1,200,000
- 2026-2027: NOI = $1,200,000 (current NOI) + $39,000 (energy savings) - $15,000 (reduced LL97 penalties) = $1,224,000
- 2030-2034: NOI = $1,200,000 (current NOI) + $39,000 (energy savings) - $33,000 (reduced LL97 penalties) = $1,206,000
- 2035+: NOI = $1,200,000 (current NOI) + $39,000 (energy savings) - $33,000 (reduced LL97 penalties) = $1,206,000

11.4 NOI Visualization Over Time

The following chart shows how NOI evolves differently in upgrade vs no-upgrade scenarios across LL97 compliance periods:



This NOI comparison reveals the true financial impact of upgrade timing:

- **Immediate NOI Boost**: Upgrading in 2024 increases NOI by $174K/year (17% increase)
- **Penalty Avoidance**: Without upgrade, NOI drops $150K in 2030+ due to higher penalties
- **Perpetual Benefit**: The $306K annual NOI gap continues indefinitely post-2035, creating sustained property value advantages
- **Risk Mitigation**: Upgrading protects against declining property values from ongoing LL97 penalties
- **Optimal Window**: Early upgrade (2024-2027) captures maximum BE Credit and avoids penalty escalation
- **Long-term Value**: LL97 compliance continues post-2035, making upgrade benefits permanent rather than temporary

12 Property Value Analysis

Property values are calculated by dividing NOI by the cap rate, providing a direct correlation between operational performance improvements and asset valuation.

12.1 Property Value Calculation Functions

Property value calculations use the same scenarios as NOI but apply cap rate conversion:

```
function calculatePropertyValueNoUpgrade(year: number, capRate: number = 0.04): number {
  const adjustedNOI = calculateAdjustedNOINoUpgrade(year); // NOI minus actual LL97 penalties
      paid
  return adjustedNOI / capRate;
}

function calculatePropertyValueUpgrade(year: number, capRate: number = 0.04): number {
  const adjustedNOI = calculateAdjustedNOIUpgrade(year); // NOI plus energy savings minus reduced
      LL97 penalties
  return adjustedNOI / capRate;
}
```

The default cap rate of 4% reflects typical NYC multifamily properties[14].

12.2 Property Value Impact Analysis

Using a 4% cap rate typical for NYC multifamily properties:

No Upgrade Property Values

$$\texttt{propertyValueNoUpgrade} = \texttt{noiNoUpgrade} \div 0.04 \tag{57}$$

With Upgrade Property Values

$$\texttt{propertyValueWithUpgrade} = \texttt{noiWithUpgrade} \div 0.04 \tag{58}$$

Net Property Value Impact

$$\texttt{netPropertyValueGain} = \texttt{propertyValueWithUpgrade} - \texttt{propertyValueNoUpgrade} \tag{59}$$

12.2.1 Implementation Example

For a 100,000 sq ft rental building with current NOI of $1,200,000:

Without Upgrade: Building faces ongoing LL97 penalties starting in 2026

- 2024-2025: Property Value = $1,200,000 (NOI) ÷ 0.04 (cap rate) = $30.0M
- 2026-2029: Property Value = $1,050,000 (NOI) ÷ 0.04 (cap rate) = $26.25M
- 2030-2034: Property Value = $900,000 (NOI) ÷ 0.04 (cap rate) = $22.5M
- 2035+: Property Value = $900,000 (NOI) ÷ 0.04 (cap rate) = $22.5M

With PTHP Upgrade: Upgrade completes in 2025, value increases in 2026

- 2024-2025: Property Value = $1,200,000 (NOI) ÷ 0.04 (cap rate) = $30.0M
- 2026-2027: Property Value = $1,224,000 (NOI) ÷ 0.04 (cap rate) = $30.6M
- 2030-2034: Property Value = $1,206,000 (NOI) ÷ 0.04 (cap rate) = $30.15M
- 2035+: Property Value = $1,206,000 (NOI) ÷ 0.04 (cap rate) = $30.15M

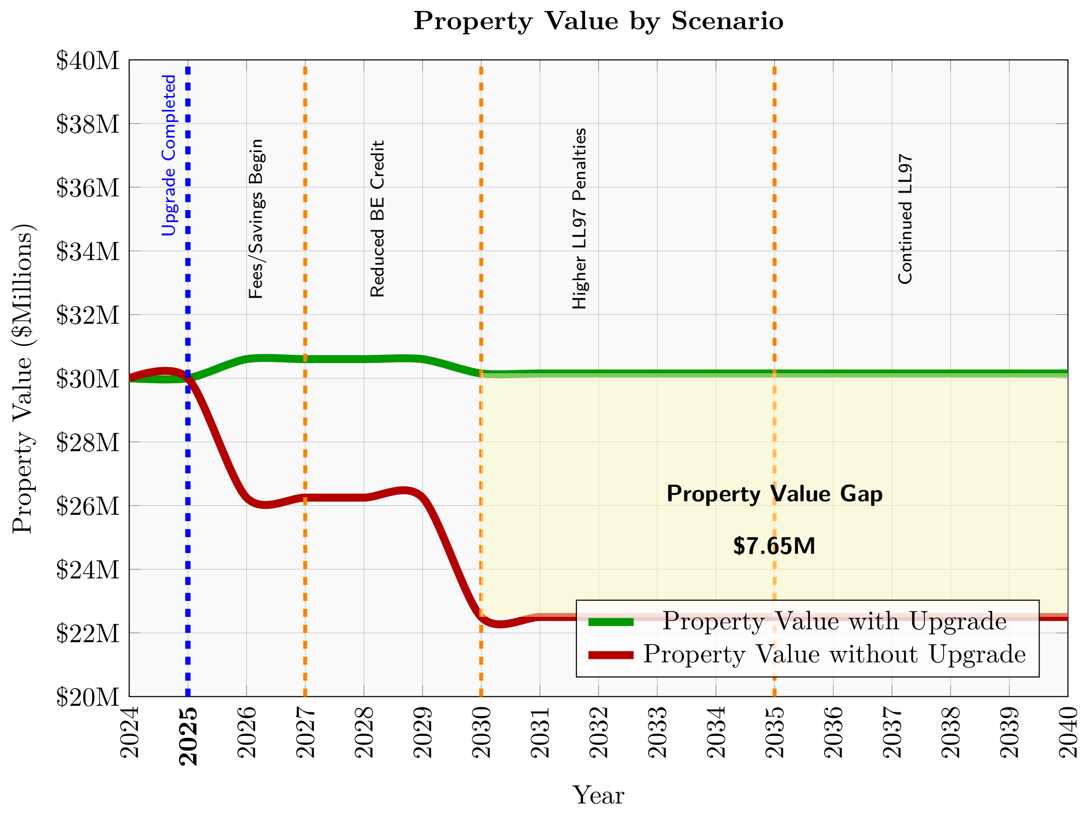
Net Property Value Gain: $30.15M - $22.5M = $7.65M

This $7.65M property value increase far exceeds typical retrofit costs of $500K-$2M, demonstrating the compelling financial case for upgrading.

[14]Cap rate of 4% used for NYC multifamily properties. This parameter can be adjusted based on market conditions, property characteristics, and investment criteria.

12.3 Property Value Impact Visualization

The following chart demonstrates property value trajectories for both scenarios:



This visualization demonstrates the substantial and sustained property value advantage of upgrading, with the $7.65M gap representing the permanent value creation from PTHP installation.

13 Implementation Notes

All variables in the codebase follow camelCase naming convention with the appropriate PTAC or PTHP suffix for clarity. The calculations are implemented in the backend services using TypeScript, ensuring type safety and consistent naming across the platform.