Fugitive Emissions Abatement Simulation Testbed

FEAST v1.0

Guide & technical documentation

Chandler Kemp[a], Adam R. Brandt[a]*

[a]Department of Energy Resources Engineering, Stanford University

*Corresponding author: abrandt@stanford.edu, +1 (650) 724 8251

Oct. 28, 2015

Contents

List of Figures

List of Tables

Nomenclature

Abbreviations

AIR	Automated IR camera LDAR program
API	American Petroleum Institute
ARPA-E	Advanced Research Project Agency-Energy
CAQCC	Colorado Air Quality Control Commission
cc	carbon-climate
cfm	cubic feet per minute
CH_4	methane
D	day
DD	Distributed detector LDAR program
EDF	Environmental Defense Fund
EPA	Environmental Protection Agency
FEAST	Fugitive Emission Abatement Simulation Testbed
FID	Flame Ionization Detector
FOV	Field of view
GWP	Global Warming Potential
IPCC	Intergovernmental Panel on Climate Change
LDAR	Leak Detection and Repair
LHV	Lower Heating Value
MIR	Manual IR camera LDAR program
mph	miles per hour
NOAA	National Oceanic and Atmospheric Administration
NR	Not recorded
Pa	Pascal
ppmv	Parts per million by volume
scfm	Standard cubic feet per minute
TCF	trillion cubic feet
TIF	Technology initialization function
tVOC	metric tonnes of volatile organic compounds
VOC	Volatile organic compound
y	year

Symbols

β	Binary function
η	Ratio of the number of wells simulated by FEAST to the number of wells supported by a capital investment

Γ Concentration path length
Λ Ray imaged by a pixel
$\mathbb{Z}$ A set of indexes
μ Mean value
Φ Concentration of effluent in a plume model
ρ Density
σ Standard deviation
τ Tortuosity: ratio of driving distance between wells to straight line distance between wells
θ Angle
$\tilde{\beta}$ Periodic binary function
$\vec{r}$ Position relative to the center of a well site
A Area
C Cost
d distance
F Buoyancy flux parameter
g Strength of gravity
H Height of a leak source in a plume model
L Leakage
M Molar mass
N Dimensionless number
n Number of samples
P Probability
Q Mass flux of a leak source
R Rate
s Stability class
T A period of time
u wind speed
V Value
v speed
W Width
x Space coordinate in a plume model measured parallel to the wind
y Space coordinate in a plume model measured orthogonal to the wind in a horizontal direction
z Space coordinate in a plume model measured vertically from the ground
NPV Net present value

Subscripts
δt referring to timesteps in a simulation
λ Lifetime
μ mean
Σ Total
A Ambient
CL Carbon Limits

D driving
E Effluent
E Equipment
F Found
g Gas
I Inspection
L Variable referring to leaks
Loc Location
N Variable referring to nonleaking components
O Operating
P Personnel (for calculating labor costs)
Q Points used in a quadrature approximation of a definite integral
R Repair
RD Real discount rate
S Survey
SI Survey interval
SU setup
T Technology
W General variable applied to all wells
WP Well pad
WWP Wells per well pad
0 Position of a leak source
LI Location time interval
M midline of a plume
s/W detectors per well

Terms

Chemical plume A plume that is buoyant due to the molar mass of its constituents

no repair NPV The NPV of an LDAR program in comparison to a scenario in which no leaks are repaired

null NPV The NPV of an LDAR program in comparison to the null program

state The set of leaks that exist in a simulated natural gas field

thermal plume A plume that is buoyant due to its temperature

1 Introduction

1.1 Motivation

Many policymakers and engineers have suggested natural gas as an affordable energy resource with relatively low GHG emissions. Indeed, natural gas contains just over half as much carbon as coal on a lower heating value (LHV) basis and can be converted to electricity more efficiently than coal [32]. However, upstream methane (CH_4) emissions from natural gas production and transportation infrastructure significantly increase the life cycle GHG emissions of natural gas. The impact on the climate of the leaks is largely due to the global warming potential (GWP) of methane: Over a 100 year time period, the GWP of fossil CH_4 is 30 ± 12 times greater than that of CO_2 on a mass basis (if known climate feedbacks are included, the estimated GWP is even higher; see Table 1.1) [35]. If leakage is more than a few percent of produced gas, transitioning energy systems from coal or oil to natural gas may have a negative impact on the climate [2][26].

Regardless of the implications for the climate, natural gas production in the United States rose dramatically over the last decade, from 20 to 27 trillion cubic feet (TCF) [45]. This growth, coupled with the abundance of natural gas resources worldwide, provides ample motivation to seek opportunities to improve the natural gas system.

Policymakers, natural gas distributors, scientists and engineers have recognized the value of reducing fugitive emissions and have invested in improving natural gas leak detection and repair (LDAR) practices. The Environmental Defense Fund (EDF), Colorado Air Quality Control Commission (CAQCC) and others have developed estimates of both the economic and environmental value of leak detection and repair programs that are intended to guide policy decisions [15][13][42]. Simultaneously, the Department of Energy's Advanced Research Project Agency-Energy (ARPA-E) program has invested tens of millions of dollars in developing new leak detection and quantification technologies [33]. The work of these groups has generally shown LDAR programs to be marginally cost effective: the value of the gas saved may or may not outweigh the cost of the LDAR program depending on the technology used, the rate of leakage at a natural gas facility, and the realized cost of implementing the LDAR programs.

This work provides a new tool to improve the allocation of resources to LDAR programs and find cost effective ways to reduce fugitive emissions. The

Table 1.1: CH_4 GWP reported by the IPCC[a]

GWP type	20 year GWP[†]	100 year GWP[‡]
basic CH_4[b]	84	28
basic CH_4 with cc feedback[c]	86	34
fossil CH_4 with oxidation[d]	85	30

[a] Intergovernmental Panel on Climate Change [35]
[b] GWPs calculated without including carbon cycle-climate feedbacks in estimating the climate forcing of CO_2.
[c] GWPs calculated including the effect of cc feedback on the climate forcing of CO_2.
[d] GWPs without the cc feedback effect, but including the climate forcing due to CO_2 that results from the oxidation of CH_4.
[†] The 90% confidence interval for the 20 yr GWP estimates is $\pm$30%
[‡] The 90% confidence interval for the 100 yr GWP estimates is $\pm$40%

Fugitive Emission Abatement Simulation Testbed (or **FEAST**) simulates the leakage from a natural gas field as a function of time under different LDAR programs. We define an LDAR program as a technology used for leak detection, the implementation of the detection technology, and the leak repair process. For example, a traditional LDAR program might consist of a Flame Ionization Detector (FID), an operator who periodically surveys natural gas equipment for leaks using the FID and a crew that repairs leaks once they are identified. Every step in identifying and repairing leaks is included in the definition of an LDAR program.

FEAST models leakage from a natural gas field as a dynamic process. FEAST generates an initial set of leaks distributed throughout a natural gas field and then adds new leaks to the field through time. The resulting CH_4 concentration downwind of every leak is simulated using a Gaussian plume model [43]. Based on the plume simulation, we apply detection criteria for several LDAR methods. FEAST identifies the leaks that will be detected and repaired under each LDAR program, and removes them from the set of leaks at the appropriate time. The total gas saved by the LDAR program is calculated as the time-integrated difference between the leakage in a null scenario and a scenario with the applied LDAR program. The null scenario represents the status quo: it allows a steady leakage rate through time as new leaks are produced and old leaks are repaired randomly without an explicit LDAR program. The economic value of the LDAR program is finally estimated based on the gas saved and the capital and operating costs of the LDAR program.

Technology assessment is possible with FEAST. Given the basic detection capabilities for a technology, FEAST can calculate the value of the leakage it

will find. The dynamic model naturally accounts for the frequency of LDAR surveys, and illustrates the value of reducing the rate at which leaks are created as well as increasing the rate at which leaks are found. Previous economic analyses provided a valuable snapshot of the potential of LDAR systems as they could be implemented today. This work builds on those results by exploring how the programs could be optimized and improved in the future. In the current version of FEAST, only leaks from well sites are modeled. Future versions of FEAST could include leaks from processing or distribution systems.

FEAST is described in the following four chapters: Literature review, Methods, User guide, and Example results.

2 Literature review

FEAST is built on decades of work by scientists and engineers to measure leakage from natural gas fields, quantify the performance of LDAR programs, and accurately model air pollution dispersion. Data published in these three fields are introduced in the following sections of the literature review. FEAST uses the data presented here to calculate the performance and value of LDAR programs.

2.1 Natural gas leakage documentation

Tens of studies have characterized natural gas leakage during the last two decades (e.g. [1][6][29]). Methods range from on-site device measurements to aircraft-based mass balance techniques. On-site studies report a broad range of leakage rates from a variety of gas fields. A low estimate of 0.42% of gross gas production was found in a sampling of 190 sites across the US [1]. A high estimate of 6.2-11% of production leaked was found in the Uintah basin [29]. At the national scale, the Environmental Protection Agency's (EPA) GHG inventory reports that an average of 6.6 Tg of CH_4 were emitted from natural gas systems annually in 2009-2012 [27]. The Energy Information Administration reports that natural gas marketed production was 465 Tg/year during the same period [45]. Together, the numbers imply that the CH_4 leaked from natural gas infrastructure accounts for 1.4% of marketed natural gas [45]. Experimental studies conducted independently of the EPA suggest that leakage may be higher [6]. Table 2.1 summarizes studies that surveyed natural gas wells and we describe them in detail below.

The American Petroleum Institute (API) published a series of studies beginning in 1980 that provide emission factors for natural gas components [14]. Of particular interest to this document, API document 4589 published fugitive emissions data from four onshore production sites, including 83 gas wells and associated equipment [5]. Every component at the sites was screened with a FID, and any component associated with a CH_4 concentration above 10 parts-per-million on a volume basis (ppmv) was identified as an emitter. A total of 1513 emitters were identified and the flux from 92 of those emitters was measured directly using a bagging technique.

API 4589 developed correlation equations relating the peak concentration and the size of leaks. The correlation equations are still used today (see [28]

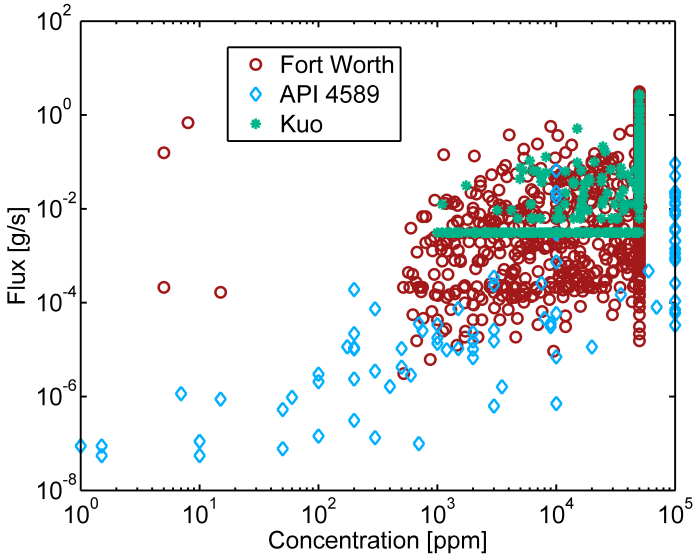

Figure 2.1: Leak flux versus measured CH_4 concentration. The vertical lines in the datasets at 5×10^4 and 1×10^5 ppmv are at the upper limits of the detection devices used in the studies. The horizontal line in the Kuo dataset is at the lower limit of the high-flow sampler used in that study. The original Kuo and Fort Worth datasets included data points that were set to default values due to limitations of the measurement tools. Those data are not included here.

and [39] for example) although the correlation between concentration and flux is known to be poor [17][31]. By combining data from three on site measurement campaigns in Figure 2.1, we show that a single concentration measurement can be caused by a flux range of over six orders of magnitude [5][12][31]. As a result of these uncertainties, only direct leak rate measurements using a high-flow sampler or equivalent technique were included in the leak distributions generated for FEAST.

A collaborative study managed by Roger Fernandez for the EPA provided a thorough characterization of leakage from five natural gas plants and 12 well sites [17]. In the study, every component in each gas plant was surveyed for leaks using a bubble test and/or gas detector. Each time the bubble test identified a leak, the maximum concentration near the leak would be measured. Only leaks that produced a concentration above 10,000 ppmv were recorded and their rate of emission quantified using a high-flow sampler. In addition to quantifying the leaks, Fernandez provided an estimate of the cost to repair each individual leak that included both direct costs of purchasing the replacement parts and paying for labor, as well as indirect costs due to lost revenue during repair.

An extensive study was conducted for the City of Fort Worth by Eastern Research Group and Sage Environmental Consulting in 2009-2011 [12]. The study was commissioned in order to estimate the impact of natural gas production on air quality in the city of Fort Worth. A total of 2126 emission points were

identified at 388 sites. The sites included 1138 wells, 8 compressor stations, and one each of the following: processing facility, saltwater treatment facility, drilling operation, fracking operation, completion operation (flowback). The entirety of each site was surveyed with an IR camera. Any leaks detected with the camera were then reviewed with a Toxic Vapor Analyzer (TVA) to find a peak concentration, and the flux was quantified using a high-flow sampler. Approximately 10% of the components were surveyed with the TVA independently of the IR camera survey in order to provide a means to estimate the difference in performance of the two techniques and ensure that the IR camera survey was sufficiently thorough. Any components with a nearby CH_4 concentration above 500 ppmv were classified as leakers and further quantified using the high-flow sampler. The TVA survey found four times as many leaks as the IR camera survey on a per component basis but the IR camera located located 60% of the volume of leakage nonetheless.

Allen et al. published a study in 2013 that quantified CH_4 leaks from natural gas production throughout the US [1]. A total of 292 gas wells were surveyed for leaks using an IR camera and 769 CH_4 sources were identified. Each leak was quantified using a high-flow sampler. Allen et al. found approximately one-half as many leaks per well as the Fort Worth TVA survey.

Another publicly available leak inventory was published by Kuo in 2012 [31]. The California Energy Commission sponsored a study to develop California-specific CH_4 emission factors for the natural gas industry in 2012. The work screened 172 wellheads for leaks using laser absorption spectroscopy to survey components from a distance. The method found 59 leaks. The leak flux was measured using either a bagging technique or a high-flow sampler.

In addition to these publicly available sources, gas detection companies are known to compile proprietary databases of the leaks that they repair. Although the data are not available to the public, Carbon Limits was given access to two such companies' datasets in an anonymous form and published their analysis of 4293 surveys of natural gas facilities comprising 39505 leak sources [42]. Although the data underlying the Carbon Limits report are not available, parts of their analysis will be used to inform the leakage model developed in FEAST.

2.2 Leak detection and repair methods

For many years the standard leak detection practice has been EPA method 21: Determination of Volatile Organic Carbon Leaks [16]. EPA method 21 requires that components be surveyed using any portable instrument that can measure the volatile organic carbon (VOC) concentration near each component with sufficient accuracy. Leaks are identified as those components with a nearby gas concentration above a threshold to be specified uniquely for each application. FIDs are one of the most common technologies used in EPA method 21.

Table 2.1: Summary of results from leakage studies of natural gas production facilities

Name	Year	Detection method	Number of wells	Number of leaks	Reference
Carbon Limits†	2014	IR camera	≈ 5300	NR	[42]
Fort Worth‡	2011	FID/IR camera	1138	2126	[12]
Allen et al.	2013	IR camera	292	769	[1]
Kuo	2012	Spectroscopy	172	59	[31]
API 4589	1993	FID	82	1513	[5]
Fernandez	2006	Bubble test	12	132	[17]

† Carbon Limits reported the number of well sites and well batteries surveyed. We estimate the number of wells by assuming an average of 3 wells per survey in the well sites and well batteries category. The number of leaks from natural gas well sites and well batteries was not published. There were 39505 leaks recorded in all facilities.

‡ All components were surveyed with an IR camera. 10% were also surveyed with a FID.

Typically FIDs measure CH_4 concentrations ranging from less than 100 ppmv up to 50,000 ppmv. 10,000 ppmv is often used as the threshold leak definition.

EPA method 21 does not provide guidance for measuring the flux from a leak [16]. If accurate flux measurements are required, a high-flow sampler is typically used.

FID surveys, coupled with high-flow sampler flux measurements, are thorough and accurate. However, the surveys are also time consuming. There is no generally accepted speed at which an FID survey can be completed, but several authors have published estimates: Fernandez uses a screening rate of 150 components per person per hour, and the CAQCC used a similar value of 120 components per hour [17][13]. The cost of surveying components at that rate can be prohibitive.

IR cameras enable natural gas facilities to be surveyed much more quickly than FID methods, although there is an enormous range in published speed estimates. Fernandez reports that the IR camera is approximately three times faster than FID methods, while the CAQCC assumed (without data) that the IR camera would provide a factor of two speed improvement [17][13]. Robinson reports surveying 2000 components per hour using an IR camera, or approximately 12 times faster than a FID survey [40]. An acceptable standard for the manual IR camera survey speed has not been established.

2.3 Plume simulation

Plume dispersion studies have been motivated by a variety of questions over the last century, ranging from how crops will be impacted by air pollution from nearby power plants to how high a smoke stack must be in order to meet

ground level air pollution standards [4][47]. FEAST requires a plume dispersion model of a particular type that has not been explicitly developed: the dispersion of CH_4 after it leaks from a faulty component. Unlike air pollution models that simulate dispersion over distances of 100 meters to tens of kilometers, FEAST focuses on dispersion within ten meters of the leak. We adapted the results from air pollution modeling to create the CH_4 dispersion simulation used in FEAST. Here, we discuss plume dispersion modeling methods published in the literature.

Developing an air pollution model is challenging because the behavior of a plume is highly dependent on ambient conditions and the nearby landscape. A compromise must be made between precision and applicability.

In the 1960s, Pasquill stability classes emerged as the standard method to incorporate atmospheric variability into plume dispersion models [22]. Pasquill initially proposed stability classes as a simple means to estimate the location and value of maximum ground level pollutant concentrations downwind of a source [37]. Pasquill suggested six stability classes based on the wind speed, insolation, and time of day. Pasquill went on to provide an example of how to use the classes to estimate the concentration of pollutants as a function of position in space, and other authors have since used the classes to develop their own models (e.g. [30]). Pasquill's stability classes have proven to be so useful that they remain standard to this day [22][43][4].

Pasquill stability classes are often applied in Gaussian plume models [43]. In a Gaussian plume model, the concentration profile of the plume is assumed to form a Gaussian surface in the plane that is normal to the wind direction. The width of the plume at a particular distance downwind from the leak source depends on atmospheric conditions. In his original work Pasquill stated that wind distribution data measured on site is required to create on accurate model [37]. However, in the absence of such data Pasquill published what he called "tentative estimates of vertical and horizontal spread" of a dissipating plume [37]. Pasquill's estimates gave the radial distance from the axis of a dissipating plume to the point where the concentration fell to 10% of the concentration along the plume axis. However, Pasquill was quick to state that even in the best cases, his estimates would only be accurate to within a factor of two [37].

Pasquill's general method for estimating the dispersion of a Gaussian plume caught on quickly. Gifford rescaled Pasquill's plume width estimates to give the spatial standard deviation of the plume concentration directly [22]. Equation 2.1 below gives the basic form of a Gaussian plume model. The standard deviation of the plume concentration is denoted by σ_z [m] and σ_y [m] [22]. Q is the leak flux [g/s], z [m] is the distance above the ground, and y [m] is the horizontal displacement orthogonal to the wind. y_0 [m] is the y position of the leak source and z_M is the vertical position of the middle of the plume

[m]. u is the wind speed [m/s]:

$$\Phi = \frac{Q}{2\pi\sigma_z\sigma_y u} \exp\left(-\left(\frac{y-y_0}{2\sigma_y}\right)^2 - \left(\frac{2(z-z_M)}{\sigma_z}\right)^2\right) \tag{2.1}$$

The curves published by Gifford for σ_y and σ_z have come to be known as the Pasquill-Gifford curves. Since then, others have followed suit by publishing their own functions for σ_z and σ_y [9][30]. Many modifications have been added to the Pasquill-Gifford curves to make them apply to buoyant plumes, plumes over variable terrain, plumes over cities, and plumes in other environments that are not captured by Pasquill's six stability classes [10].

Buoyant plume models are of particular interest here, due to the difference in density between CH_4 and atmospheric air. Most studies of buoyant plumes focus on warm stack gas emissions. In that case, standard practice is to define the midline of the plume as a function of distance downstream of the leak: $z_M = z_M(x)$. More specifically, z_M is defined by Equation 2.2, where H is the initial plume height [m], x_0 is the position of the leak source [m], u is the windspeed [m/s] and F is the buoyancy flux parameter [m^4/s^3].

$$z_M = H + 1.6F^{1/3}(x-x_0)^{2/3}u^{-1} \tag{2.2}$$

F is the rate at which the buoyancy force increases as the plume grows [N/s] divided by π times the density of the ambient air [kg/m^3]–see Equation 2.3). g is the strength of gravity [m/s^2], Q is the mass flux of effluent [g/s], ρ_E is the density of the effluent [kg/m^3] and ρ_A is the density of the ambient air [kg/m^3].

$$F = \frac{gQ}{\pi}\left(\frac{1}{\rho_E} - \frac{1}{\rho_A}\right) \tag{2.3}$$

The above two equations were developed in the late 1960s based on both theoretical analysis and data collected in tens of studies of plume rise [7]. Several seminal papers were published by Gary Briggs in 1969-1972 that combined data from many studies, compared the derived experimental plume rise equations and identified the most reliable and accurate plume rise models [8][7]. The equations (based on the final form published in 1972) are valid within a specified distance of the plume source, typically on the order of kilometers [4].

Gaussian plume models underly most prominent air pollution models today, including the EPA's AERMOD used for regulatory purposes and the National Oceanic and Atmospheric Administration's (NOAA) atmospheric dispersion model [11][21].

3 Methods

3.1 General model structure

The basic structure of FEAST is shown in Figure 3.1. FEAST integrates the total leakage through time under different LDAR scenarios to enable meaningful comparison between LDAR programs. The datasets that FEAST relies on include leakage from operating natural gas fields, recorded wind data, and published leak repair cost data. The user must provide inputs to FEAST to define the various LDAR programs and characteristics of the natural gas field. The input data and user defined variables are indicated by square boxes in Figure 3.1. Processes in FEAST are indicated by ovals.

FEAST uses data and inputs to define the set of leaks in the simulated natural gas field at an initial time. Once the natural gas field is initialized, FEAST allows the *state* of the field to evolve through time. The *state* is defined as the set of leaks that exist in the simulated natural gas field. At a specified treatment interval the detection technology is applied to the natural gas field, and the leaks that it identifies and repairs are removed from the state. New leaks are then added to the state through a random process and the simulation proceeds to the next time step. The leakage is recorded at each time step, and the total cost of the LDAR program and value of the gas saved is calculated at the end of the simulation.

In the following sections we present all of the modules within FEAST. Each section includes the following information:

- Purpose of the module
- Definition of variables in the module
- Description of equations implemented in the module
- Summary of assumptions in the module
- Table of variables including symbol, variable name, default value and sensitivity ranges

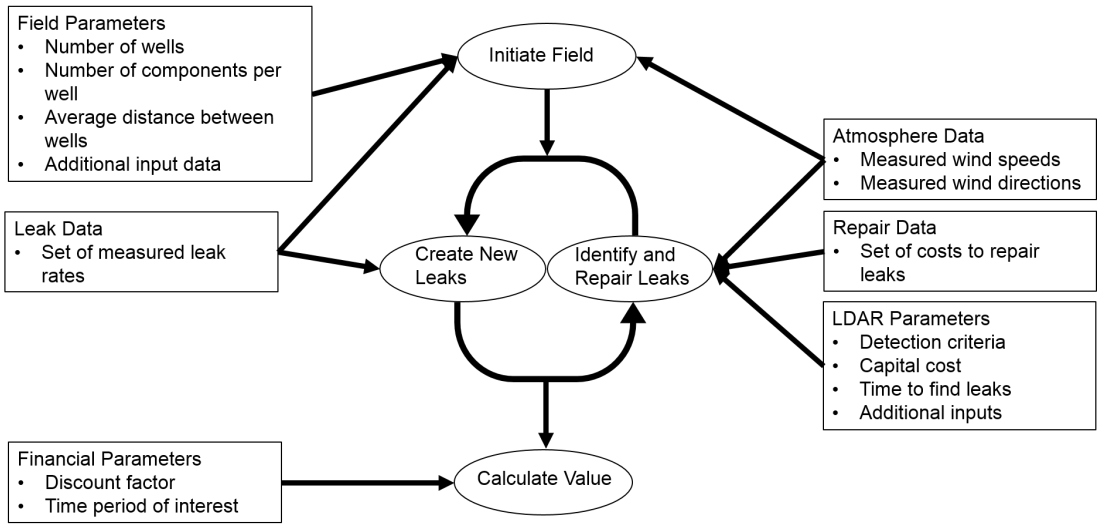

Figure 3.1: High level schematic representation of FEAST.

3.2 Natural gas field initialization

Here we describe each of the variables that must be defined to initialize a gas field in FEAST. All gas field variables are summarized in Table 3.1 at the end of this section.

3.2.1 Distance between wells

The cost of traveling between wells in a gas field is proportional to the required driving distance between wells d_W [m]. The range of values chosen for d_W was determined by measuring the density of well pads in three representative locations in the Barnett and Bakken fields (ρ_{WP} [pads/km^2]), dividing by the average number of wells per gas pad N_{WWP} [wells/pad], and then multiplying by a tortuosity factor (τ [driving distance/straight line distance]) to account for indirect driving routes between wells:

$$d_W = \sqrt{\frac{N_{WPP}}{\rho_{WP}}} \times \tau \tag{3.1}$$

The well pad density was measured by visually counting well pads in satellite imagery (for example, see Figure 3.2). τ and N_{WPP} only affect results through d_W, therefore, no sensitivity range was chosen for them, but a broad range in d_W was analyzed.

3.2.2 Gas field properties affecting LDAR survey time

Two additional gas field properties must be defined in order to model the leak detection technologies of interest here: the area surrounding a well that may harbor a leak, A_W [m^2 per well], and the number of components per well $N_{c/W}$

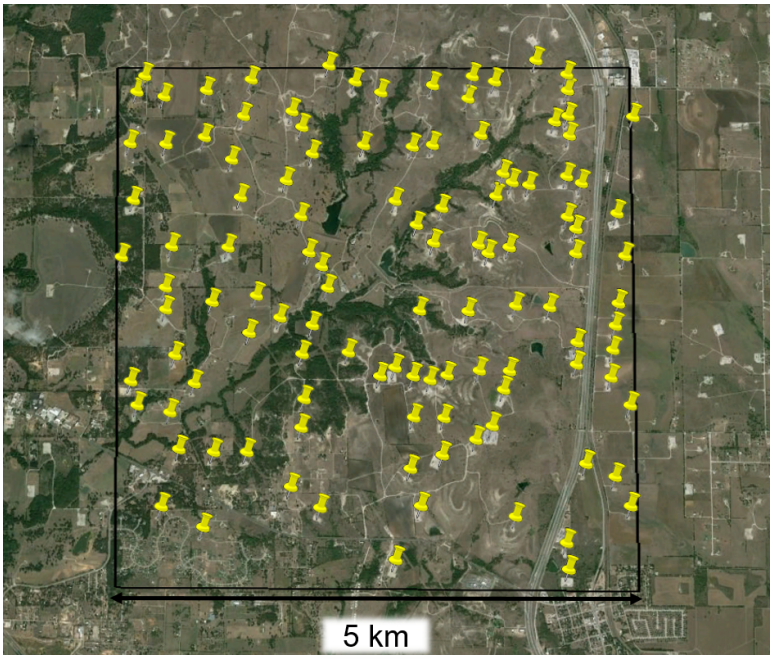

Figure 3.2: High density drilling area in the Barnett Shale of Texas (33^o 4′ 44″ N 97^o 30′ 06″ W)

[components per well]. For some LDAR technologies, the time to survey a well depends on the ground area that must be searched for leaks. In other cases, the cost depends on the total number of components per well, $N_{c/W}$ [components per well]. The area associated with each well that may harbor a leak is defined in the gas field initialization and assumed to be constant for every well. The number of components per well (including components in separators and other equipment associated with a natural gas well but not directly part of the wellhead itself) is estimated based on inventories from several sources [5][17][12].

3.2.3 Leak initialization

FEAST generates the initial set of leaks in a three step process. First, FEAST draws the number of leaks at the beginning time $N_L(t = 0)$ for each well from a normal distribution. The mean and standard deviation of the distribution are both defined as input variables; in this study values were chosen based on leak inventories published in the literature [1][17][12]. The true distribution of leaks per well is skewed (see Figure 3.3), but using a normal distribution does not affect the results of FEAST. FEAST treats each leak separately in the detection and repair process. As a result, the distribution of leaks among all of the wells does not affect the LDAR program values calculated by FEAST.

The default mean number of leaks per well is estimated using data from the Fort Worth study [12]. Equation 3.2 accounts for the two different detection methods used by the Fort Worth study. $\mu_{L/W}$ is the average number of leaks

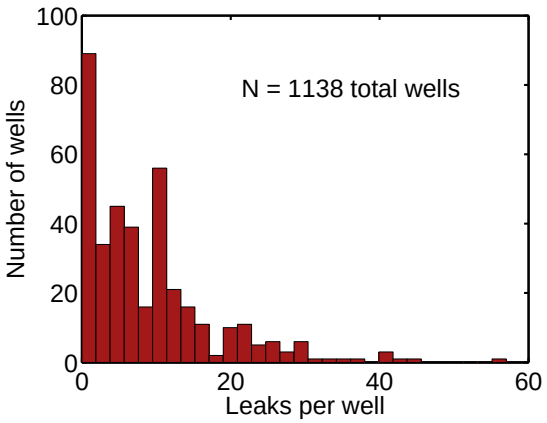

Figure 3.3: Distribution of leaks per well based on the Fort Worth dataset [12]

per well, $N_{L,FID}$ is the number of leaks detected with the FID, 10 is the ratio of total components to components surveyed with the FID, and $N_{L,IR}$ is the number of leaks detected with the IR camera.

$$\mu_{L/W} = \frac{N_{L,FID} \times 10 + N_{L,IR}}{N_W} \tag{3.2}$$

Selecting leaks from the empirical dataset to create a simulated leak distribution is done in a similar way. FEAST uses the full set of leaks found with the FID and a random sampling of 10% of the leaks found using an IR camera in the Fort Worth dataset. Finally, once the number of leaks per well and leak size distribution have been determined the location of the leaks are chosen randomly from a uniform distribution in a $10 \times 10 \times 5$ m region specified by Equations 3.3-3.5. $P()$ is a probability density function with units of [m^{-1}].

$$P(x_0) = \begin{cases} \frac{1}{10} \text{ for } -5 \leq x_0 \leq 5 \\ 0 \text{ Otherwise} \end{cases} \tag{3.3}$$

$$P(y_0) = \begin{cases} \frac{1}{10} \text{ for } -5 \leq y_0 \leq 5 \\ 0 \text{ Otherwise} \end{cases} \tag{3.4}$$

$$P(z_0) = \begin{cases} \frac{1}{5} \text{ for } 0 \leq z_0 \leq 5 \\ 0 \text{ Otherwise} \end{cases} \tag{3.5}$$

Data provided by the Fort Worth study [12] are used to produce the leak size distributions in FEAST under default settings, although the model supports using data from Allen [1] as well (See Figure 3.4). The Fort Worth and Allen studies were chosen because they offer the largest publicly available leakage datasets for natural gas production facilities.

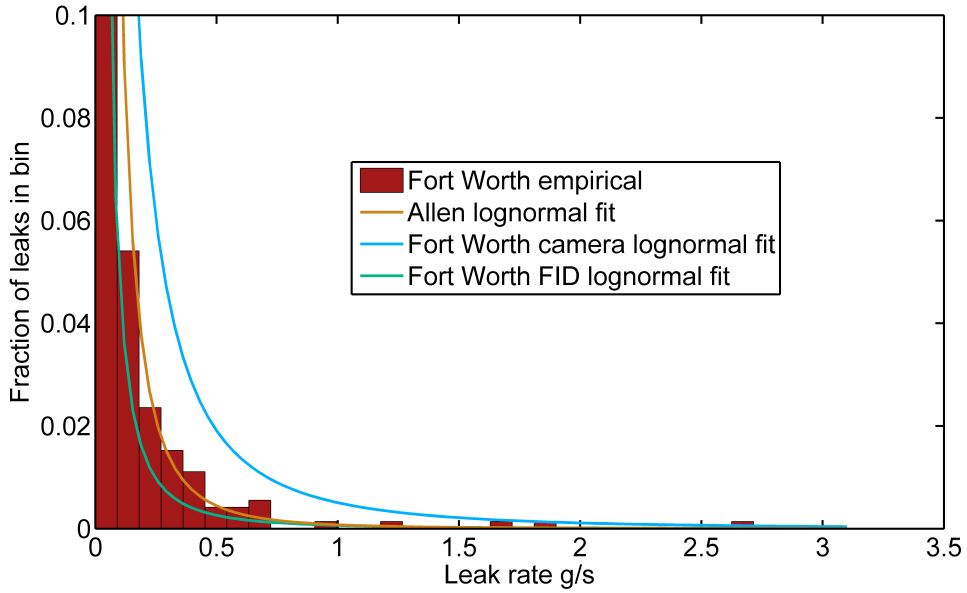

Figure 3.4: Leak size distributions from the literature. Bars indicate an example empirical leak distribution used in FEAST built from the Fort Worth datasets. Lines indicate lognormal fits to leak size distributions from three leak datasets used by FEAST [12][1].

3.2.4 Number of wells

The final variable to be defined in the gas field initialization is the total number of natural gas wells N_W. In general, the results from FEAST are given on a per well basis so that they are independent of the size of the simulation in the model. However, N_W affects the performance of FEAST: increasing N_W in the simulation reduces the stochastic variability between runs, while decreasing N_W allows for shorter run times. The user can choose a value of N_W by compromising between model variability and run time.

Table 3.1: Gas field initialization variables

Symbol	Name	FEAST Variable*	Units	Default value	References
A_W	Survey area per well	GasField.WellArea	m^2	100	-
$\tau^\dagger$	Tortuosity	-	ratio‡	1.7	-
d_W	Average driving distance	GasField.SiteSpacing	m	700	Equation 3.1
H_W	Maximum leak height	GasField.H0Max	m	5	-
$N_{c/W}$	Components per well	GasField.ComponentsPerSite	-	650	[12][5][17]
$\mu_{L/W}$	Average leaks per well	LeaksPerWell	-	6	[12][1][17]
N_W	Number of wells	GasField.SiteCount	-	100	-
$N_{WPP}{}^\dagger$	Wells per well pad	-	-	2.8	[12]
$P(Q)$	Leak size distribution	GasField.LeakData	-	[12]	-
R_L	Leak production rate	GasField.LeakProductionRate	$\frac{\text{leaks}}{\text{component-day}}$	10^{-5}	[42]
R_{R-Null}	Null repair rate	GasField.NullRepairRate	$\frac{\text{repair}}{\text{leak-day}}$	1×10^{-3}	Equation 3.7
$\rho_{pads}{}^\dagger$	Density of well pads	-	pads/km^2	2.0	[24][25][23]

* All variables are defined in the file `InitializationFunctions/GasFieldInitiator.m`.
† Sensitivity in this variable is included in the driving distance sensitivity.
‡ Tortuosity is defined here as the dimensionless ratio of driving distance [m] to interwell spacing [m].

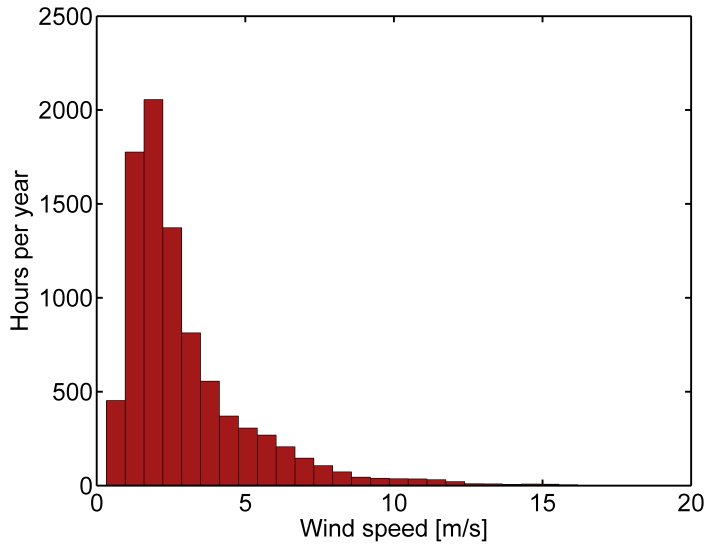

Figure 3.5: Wind speed distribution from MONITOR wind dataset [33].

3.3 Atmosphere Initialization

In addition to an initial set of leaks, atmospheric variables must be set. Under default settings FEAST uses wind data provided by the ARPA-E MONITOR challenge, see Figure 3.5 [33]. From the wind dataset, a random wind speed is chosen and stored for each time step in the simulation. A wind direction dataset is generated in the same way using wind direction data from Fort Worth, TX, see Figure 3.6[1] [36]. Wind direction data are chosen from a different dataset because the MONITOR wind data are not directional.

The stability class for each time step is chosen randomly, within the wind speed constraints provided by Table 3.2 [43]. For example, if the wind speed for a particular time step is greater than 6 m/s, the stability class is randomly chosen to be C or D with equal probability.

[1] Wind rose created using a script by D. Pereira [38].

Table 3.2: Stability class dependence on wind speed [43]

Windspeed (m/s)	Possible Daytime Stability Classes
<2	A or B
2	A, B, or C
4	B or C
6	C or D
>6	C or D

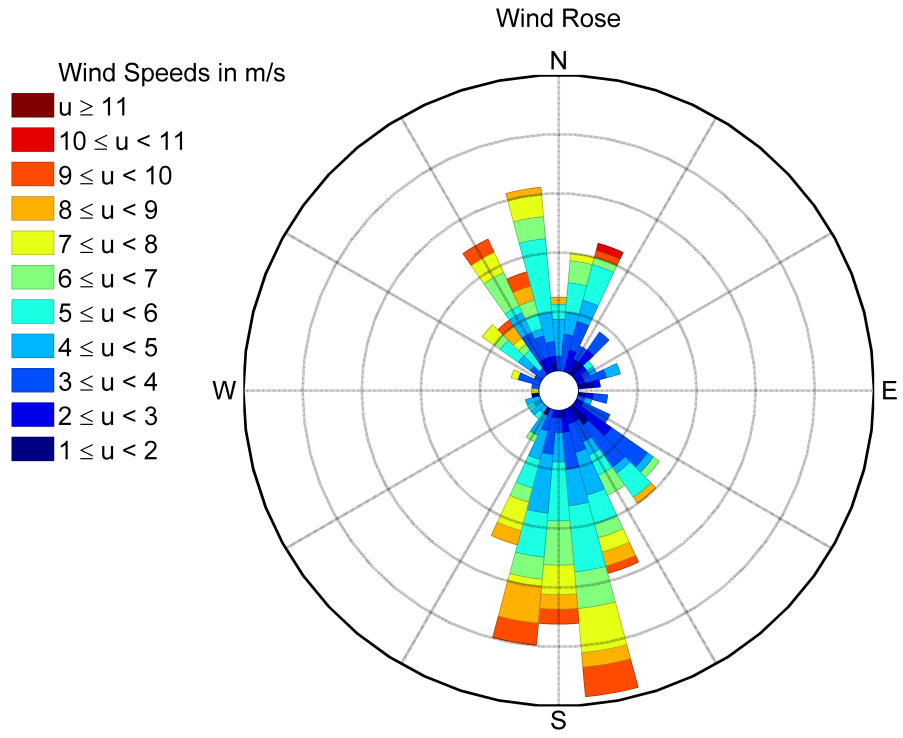

Figure 3.6: Wind rose for Fort Worth, TX weather data [36].

Table 3.3: Atmosphere probability distributions

Symbol	Name	FEAST Variable*	Sources
$P(u)$	Set of wind speeds	AtmosphereStruct.u	[33]
$P(\theta)$	Set of wind directions	AmtosphereStruct.theta	[12]

* All variables defined in the file `InitializationFunctions/AtmosphereInitiator.m`.

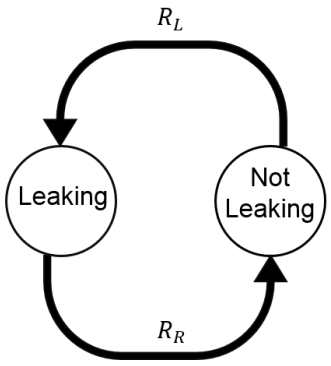

Figure 3.7: Two state Markov model

3.4 Markov model

Once the natural gas field and atmospheric modules have been initialized, the simulation begins. Two fundamental assumptions allow the natural gas field to be simulated using a basic, two state Markov model [20]:

- every component in the natural gas field can be in one of two states (leaking or not leaking)
- at each time step there is some probability that a component will change state independent of its history

The model is represented graphically in Figure 3.7, where R_R is the rate at which leaks may be repaired, and R_L is the rate at which leaks are created.

3.4.1 Model implementation

The numerical model implemented here is an approximation of Equation 3.6. In Equation 3.6 $N_N(t)$ is the number of nonleaking components as a function of time (t), $N_L(Q,t)$ is the number of leaking components as a function of the leak size (Q) and the time (t), R_L is the rate at which nonleaking components begin leaking [new leaks per nonleaking component per day] and R_R is the rate at which leaking components are repaired [repaired leaks per leak per day]. The Markov model variables are summarized in Table 3.4.

$$\frac{dN_L(Q,t)}{dt} = N_N(t)R_L(Q) - N_L(Q,t)R_R(Q,t) \tag{3.6}$$

The definition of a leaking component is arbitrary. In principle every component could be assigned a leakage rate greater than or equal to zero, but the vast majority of components would produce such small amounts of natural gas that they would have no significant impact on the results. Rather than be concerned with leaks too small to detect, we developed a model that reflects the available data: FEAST randomly generates a number of new leaks at each

Table 3.4: Markov model variables

Symbol	Name	FEAST Variable*	Units
N_N	Number of nonleaking components	-	components
N_L	Number of leaking components	GasField.LeakCount	components
N_c	Total number of components	GasField.ComponentCount	components
$R_L(Q)$	Rate that leaks are formed	-	leaks/component-day
$R_R(Q)$	Rate that leaks are repaired	-	repairs/leak-day

* All variables defined in the file `InitializationFunctions/GasFieldInitiator.m`.

time step, and then assigns a flux to each leak from the distribution of leak sizes $P(Q)$ defined in Section 3.2.3.

The challenge for FEAST is to accurately represent R_L and R_R. FEAST classifies R_L as a property of the natural gas infrastructure itself; it is independent of any LDAR program. Conversely, R_R is calculated independently for each LDAR program.

We also assume that for each LDAR program R_R is the sum of a repair rate due to the applied technology and a "null" repair rate that is inherent to the natural gas field. The null repair rate is due to operators noticing and repairing leaks during normal work practice without an explicit LDAR program. The null repair rate allows the amount of leakage in the natural gas field to reach a steady state even without a LDAR program.

The effect of R_L is simulated in FEAST by introducing new leaks at every time step. We assume that leaks are produced at the same rate throughout time. Furthermore, since the number of nonleaking components never varies by more than 3% in the simulations, we approximate N_N as constant in time as well. Therefore, the rate at which leaks are produced is constant throughout the simulation.

$$R_{R-Null} = \frac{R_L N_N}{R_N} \tag{3.7}$$

3.4.2 Determination of Markov variables

Each leak measurement campaign gives a measure of N_N and N_L at a particular instant in time, as well as the distribution of the sizes of existing leaks. The measured values of N_N and N_L are the result of a real-world Markov process that follows Equation 3.6. In order to estimate R_R and R_L we need to know N_N and N_L as functions of time.

We have two methods for estimating R_L based on the data that are available. The first method is to consider the total leakage observed in the Fort Worth study, and divide by the average age of wells in the region at the time of the study. The average leakage rate in the Fort Worth study was 0.3 g/s per well. We do not have data regarding the age of the wells in the Fort Worth

study so we assume that they match the average age of wells in the Fort Worth Basin. Based on the year drilling permits were issued by the Texas Railroad Commission, we estimate the average age of wells to be 4.5 years at the time of the Fort Worth survey. Thus we estimate a leak creation rate of 1.8×10^{-4} g/s per day per well.

There are many sources of error in the above estimate that are difficult to quantify. The first is the amount of leakage that exists in new gas wells. We calculated the rate by assuming that the leakage in new wells is negligible in comparison to the leakage that accumulates over time. If most of the leakage in the Fort Worth wells was built into the original infrastructure, then the above estimate exaggerates the rate at which leaks grow. Conversely, we have omitted any null repair rate in the above calculation. Over the years that the wells have been in existence, some leaks have surely been repaired even in the absence of an LDAR program. This likelihood implies that our above calculation would underestimate the rate of leak production.

Data from Carbon Limits [42] reduces the sources of error mentioned above. The Carbon Limits study included results from 427 annual surveys of natural gas facilities, and estimated the value of performing leak detection and repair surveys at different time intervals. Carbon Limits reports that in the first year following a leak detection and repair survey, 1.8 tVOC are emitted from the average natural gas well battery[2]. Assuming that the leakage rate increases linearly from zero throughout the year, and estimating the mean VOC to gas ratio, we calculate the necessary leak production rate to be 3.8×10^{-4} g/s per day per well (see Section A.2 for further discussion).

Taking these two results into consideration, the default leakage production rate in FEAST is set to 2.6×10^{-4} g/s per day per well. We divide the leakage production rate by the number of components per well and the average leak size to find the leak production rate R_L in units of leaks/day per component. We then defined the average number of new leaks at each time step according to Equation 3.8, where δt is the length of the time step [days] and N_c is the total number of components in the simulation:

$$\delta N_L = N_c R_L \tag{3.8}$$

FEAST generates a random number of new leaks at each time step with an average of δN_L. The size of the leaks is again chosen randomly from the same $P(Q)$ used to create the initial set of leaks (see Figure 3.4).

The rate at which leaks are found and repaired is more complex because it depends on the LDAR program being modeled in a given simulation. As a result, a separate module must be simulated for every LDAR program. Each of these modules is discussed separately in Section 3.6. However, the null repair process and associated repair rate are consistent in all of the LDAR program modules.

[2]A well battery is defined here as one or more natural gas production wells and associated equipment

Given the previously estimated value of R_L, the number of leaks observed in a particular survey may be the result of one of three scenarios. The leakage may have reached a steady state, in which case we can compute R_{R-Null} from the equality $R_{R-Null}N_L = R_L N_N$. Alternatively, the number of leaks may be growing if $R_{R-Null}N_L < R_L N_N$, or the number of leaks may be be declining if $R_{R-Null}N_L > R_L N_N$. The true scenario will depend on the age of the natural gas field, the number of leaks present in the field when it was first constructed, and the null repair rate associated with that particular field.

In the default settings, we assume that the Fort Worth field was in a steady state during the Fort Worth air quality study, and compute R_{R-Null} based on Equation 3.7. Though little data is available on this parameter, we find our assumption to be the most reasonable in the absence of further evidence.

3.5 Gaussian plume model

A Gaussian plume model is used to estimate CH_4 concentration downwind of leaks (see Equation 3.10) [43]. The concentration as a function of position is a product of Gaussian functions. A basic Gaussian function is provided in Equation 3.9.

$$g(\xi) = \exp(\frac{-\xi^2}{2}) \tag{3.9}$$

In the Gaussian plume model, ξ is replaced by y and z coordinates shifted to account for the position of the source of the plume and rescaled to account for the plume width, as shown in Equation 3.10.

$$\Phi = \frac{Q}{2\pi u \sigma_y \sigma_z} g\left(\frac{y-y_0}{\sigma_y}\right)\left[g\left(\frac{z-z_M(x)}{\sigma_z}\right) + g\left(\frac{z+2z_M(x)}{\sigma_z^2}\right)\right] \tag{3.10}$$

The coordinates x, y, and z are defined as shown in Figure 3.8: the origin is at ground level at the center of the well pad, x is measured in the direction of the wind, y is measured perpendicular to the wind, and z is the distance above the ground [m]. x_0, y_0, and H are the x, y, z coordinates of the leak source [m] (other variables are defined in Equation 2.1).

Each term in Equation 3.10 has a simple physical interpretation. The first term shows that the concentration everywhere is proportional to the size of the leak and inversely proportional to the windspeed. The first Gauss function shows that the modeled plume has a Gaussian profile in the y direction. The two final Gauss functions show that the plume has a Gaussian concentration profile in the z direction, but there is a mirror image of the plume superimposed on top of it to account for reflection of the plume off of the ground.

In some plume models, an additional term is included to account for absorption at the ground level as well as the possibility of reflection from an atmospheric inversion. We assumed that the amount of CH_4 from a natural gas leak abosrbed by the ground would be negligible. We further assumed that the atmosphere would be homogeneous throughout our simulation since we were primarily interested in the behavior of the plume within 50 meters of the leak source at heights below 30 meters. The stability of the atmosphere is accounted for in the calculation of σ_y and σ_z, but no other atmospheric modeling is included in FEAST.

Three of the variables in Equation 3.10 are functions of x: σ_z, σ_y and z_M. Several different functions have been published for σ_z and σ_y, e.g. [22][30]. Standard models are built using equations that give realistic results for distances over 100 meters, but they may become inaccurate at shorter length scales. For example, Figure 3.9 shows three functions for σ_z that have been published in the literature [43]. Near the leak source, σ_z actually diverges in the function attributed to Pasquill-Gifford, while σ_z for the most stable class is larger than σ_z for the least stable class in the functions attributed to Klug

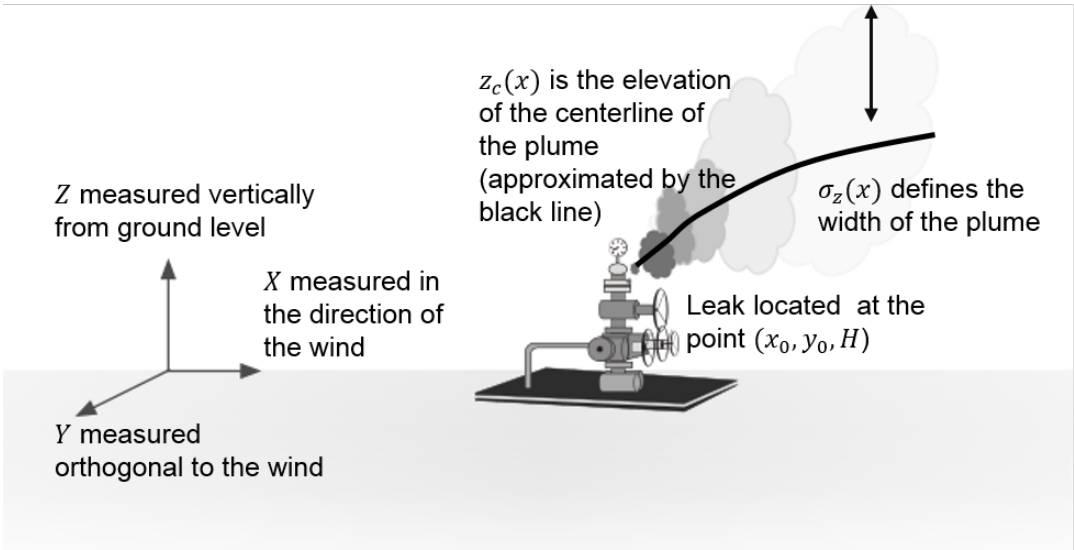

Figure 3.8: Schematic of the gaussian plume model coordinate system.

[30]. Clearly, these representations of σ_z are not physical at such short length scales. For the purposes of this study, σ_z and σ_y were simply approximated as linear functions from zero at the leak source to the values given at 100 meters downwind of the plume for each stability class by Gifford [22].

$z_M(x)$ accounts for the buoyancy of CH_4. FEAST implements the buoyant plume model presented by Beychok and described in equations 2.2 and 2.3 to compute $z_M(x)$ [4]. The plume model by Beychock is designed for a hot buoyant plume, and we expect a methane plume that is buoyant due to the molar mass of CH_4 to behave similarly (see Appendix A.3). An example concentration profile of a buoyant plume is shown in Figure 3.10. It shows that the plume has its highest concentration immediately downwind of the plume source, and the plume slowly becomes less concentrated and rises as it is carried downwind.

For simplicity, we approximate natural gas as 100% CH_4. This assumption only affects the results of FEAST through the performance of the detection technologies. Some technologies may be more sensitive to other components of natural gas, making FEAST underestimate technology performance. Other technologies may be less sensitive to non-CH_4 hydrocarbons. The effect of this assumption is explored through a sensitivity study on the performance metric of each technology.

3.6 LDAR pogram modules

FEAST provides the basic simulation structure to which different LDAR program modules can be applied. This study includes simulation of four different types of LDAR programs:

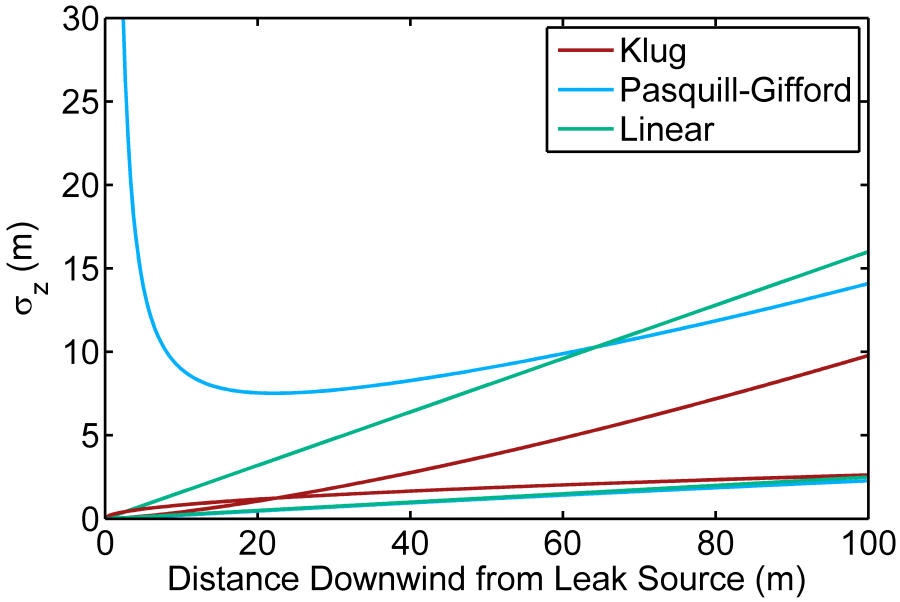

Figure 3.9: $\sigma_z(x)$ for the most and least stable atmospheric stability classes using various published functions [30][43] . Feast uses a linear fit to the values published at 100 m by [22].

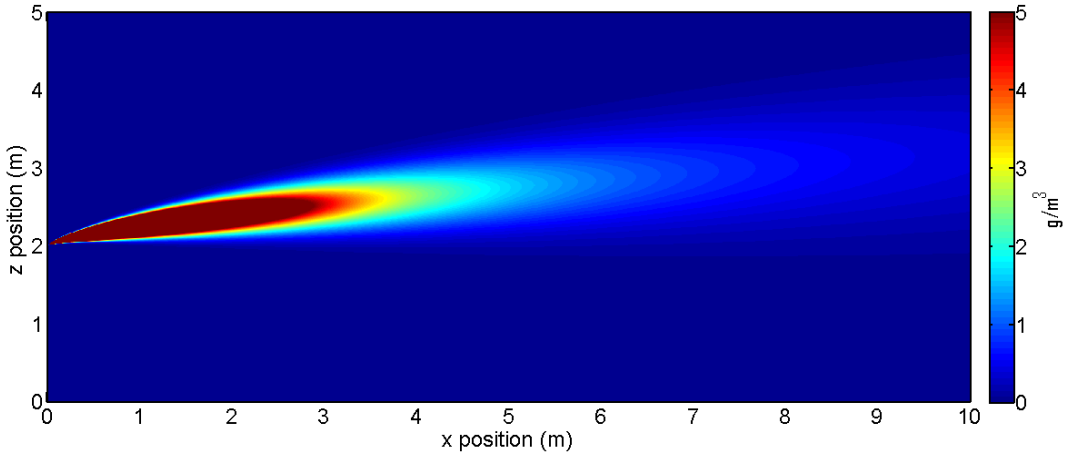

Figure 3.10: Simulated concentration profile of a natural gas leak with a gas flux of 2 g/s, wind speed of 1 m/s and stability class C.

- Traditional FID survey (FID)
- Manual IR camera survey (MIR)
- drone-mounted automatated IR camera survey (AIR)
- Distributed detector monitoring (DD)

Each LDAR program module has two components: a method to estimate the leakage that an LDAR program will detect, and a method to calculate the cost of the program. During a simulation, the signal from every leak is compared to the detection criteria of the method to determine whether or not it is detected. For a continuous monitoring technology like the distributed detector, each leak is checked at every time step. For LDAR programs based on periodic surveys, leaks are only checked once per survey period. The found leakage (L_F [g/s]) is calculated as the sum of found leakage over the set of all leaks ($\mathbb{Z}_L$), as shown in Equation 3.11. We use β to represent a binary function. Specifically, $\beta_{F,T}$ is a binary function that returns one if the leak is found, and zero otherwise (the subscript F indicates that it is a detection function). $\beta_{F,T}$ depends on the leak size (Q), wind-speed ($u(t)$) and stability class ($s(t)$), and is specific to each technology (T). The explicit form of $\beta_{F,T}$ is presented for each technology in the subsequent sections.

$$L_F(t) = \sum_{i \in \mathbb{Z}_L} Q_i \beta_{F,T}(Q_i, u(t), s(t)) \tag{3.11}$$

The cost of each LDAR program is calculated based on the capital cost of the equipment, the operating cost to search for leaks, and the cost to repair the leaks. The total cost at each time step for every LDAR program ($C_\Sigma(t)$) is calculated according to Equation 3.12 below.

$$C_\Sigma(t) = C_E(t) + C_O(t) + C_R(L_F(t)) \tag{3.12}$$

The capital cost of equipment ($C_E(t)$) includes the upfront cost of the LDAR program, as well as the cost to periodically replace equipment, as shown in Equation 3.13. The equipment cost is scaled to account for the finite size of the simulation. More specifically, if the capital investment C_{Cap} provides the means to monitor N_{max} wells, but FEAST only simulates N_{FEAST} wells, then the capital investment is multiplied by a factor $N_{FEAST}/N_{max} = \eta$ to compute the equipment cost.

$$C_E(t_j) = C_{cap} \tilde{\beta}(t, T_\lambda) \eta \tag{3.13}$$

The function $\tilde{\beta}(t, T_\lambda)$ is a periodic binary function implemented multiple times in FEAST. It returns one exactly once per period of time T_λ as shown by Equation 3.14 below:

$$\tilde{\beta}(t, T_\lambda) = \begin{cases} 1 & \text{if } mod(t, T_\lambda) < \delta t \\ 0 & \text{otherwise} \end{cases} \tag{3.14}$$

In Equation 3.14, δt is the length of a time step in the simulation. Therefore, $\tilde{\beta}$ returns one if and only if the simulation time t is between nT and $nT + \delta T$ for some n, where n is any nonnegative integer. In Equation 3.13, $\tilde{\beta}$ is used to add capital costs to the program each time the equipment lifetime T_λ is reached in the simulation. Although each technology module may in fact have multiple types of equipment with different lifetimes, FEAST only allows for one representative lifetime for each technology.

The operating cost $C_O(t_j)$ includes both maintenance costs and labor costs associated with the LDAR program. C_O for all survey based LDAR programs is calculated based on the survey operating cost equation defined by Equation 3.15. Here, C_m is the maintenance cost of the program [\$/year], C_P is personnel cost [\$/hour], T_S is the labor time required to complete a survey [hours], and T_{SI} is the survey interval [days].

$$C_O(t) = C_m \delta t + C_L T_S \tilde{\beta}(t, T_{SI}) \tag{3.15}$$

Finally, the repair costs are calculated at each time step according to Equation 3.16, where $\phi(P(C_R), N_{F,L})$ is a function that creates a set of leak repair costs by choosing $N_{F,L}$ random samples from $P(C_R)$ with replacement. $P(C_R)$ is a distribution of repair costs discussed further in Section 3.7. $N_{F,L}$ is the number of leaks that are found and repaired.

$$C_R(L_F(t)) = \sum \phi(P(C_R), N_{F,L}) \tag{3.16}$$

As outlined above, all of the LDAR program modules share many of the same features. However, the values chosen for each variable, the definition of the detection functions ($\beta_{F,T}$), and the methods used to estimate the variables presented above differ between modules. The subsequent sections present the unique characteristics of each module. Specifically, each section presents the following information:

- Basic design of the LDAR program
- Explanation of the detection criteria
- Description of cost calculations
- Table of variables

3.6.1 Null module

At each time step, the null module randomly selects leaks to remove from the natural gas field according to Equation 3.17. Here, $U(0,1)$ is a function that returns a random number from a uniform distribution between zero and one. δt is the length of a time step in the simulation, and R_{R-Null} is the null repair rate defined in subsection 3.4.2.

$$\beta_{F,Null}(Q_i, u(t), s(t)) = \begin{cases} 1, & \text{if } U(0,1) > R_{R-Null}\delta t \\ 0, & \text{otherwise.} \end{cases} \tag{3.17}$$

Unlike other detection modules, the null detection function does not depend on atmospheric conditions u and s or leak size Q_i. The null module has an equal probability of repairing every existing leak. As a result, with the default settings of R_{R-Null} and the leak production rate R_L, the number of leaks and distribution of leak sizes stays approximately constant through time.

There are no finding costs or capital costs associated with the null module because it simulates leak detection without investing additional labor or capital beyond normal operations. Repair costs are attributed to the null module in the same way that they are attributed to other modules.

3.6.2 Flame ionization detector module

FID based LDAR programs require manually surveying every component for leaks. An operator positions the FID next to each component and measures the concentration of CH_4. If a concentration above a particular threshold is detected, the component is identified as a leaker and repaired.

The detection function for the FID detector is quite simple: every leak is found (see Equation 3.18).

$$\beta_{F,FID}(Q_i, u(t), s(t)) = 1 \tag{3.18}$$

We assume that every leak generated in FEAST will also be detected by an FID survey because the smallest leaks in the datasets FEAST uses are based on empirical results from FID surveys. Furthermore, FIDs can detect very small leaks and we expect the number of leaks overlooked with the technology to be negligible.

The operating cost is calculated according to the survey operating cost equation 3.15. T_S is the sum of the inspection time (T_I), driving time (T_D), and the set up time per well (T_{SU}) 3.19.

$$T_S = (T_I + T_D + T_{SU})N_W \tag{3.19}$$

The inspection time is estimated using Equation 3.20. The FID survey rate ($R_{s,FID}$) is estimated based on survey rates published in the literature [17].

$$T_I = N_c R_{s,FID} \tag{3.20}$$

The driving time is simply estimated based on the driving distance between wells defined in section 3.2, and an assumed driving speed (v_D), as shown in Equation 3.21.

$$T_D = \frac{d_W}{v_D} \tag{3.21}$$

The initial capital cost of the FID program ($C_{0,FID}$) is the cost of purchasing an FID detector and a truck for the operator to drive from site to site.

Table 3.5: FID Variables

Symbol	Name	FEAST Variable*	Units	Default Values	Sources
C_{truck} †	Truck capital	-	$	30000	-
C_{FID} †	FID capital	-	$	5000	[17]
C_{Cap}	Total capital	Capital0	$	35000	-
C_m	Maintenance cost	Maintenance0	$/y	3500	-
T_λ	Lifetime	FIDStruct.lifetime	d	3650	-
R_S	Survey speed	processStruct.surveySpeed	components/hour	150	[17]
T_{SI}	Survey interval	processStruct.surveyInterval	d	100	-
T_{SU}	hours	Setup time	processStruct.setupTime	0.5	-
v_{LD}	Driving speed	processStruct.driveSpeed	m/s	15	-

* All variables are defined in the file `DetectionTechnologyModules/FID.m`.
† The sensitivity range of this variable is included in the sensitivity of the total capital

3.6.3 Manual IR camera module

In the MIR LDAR program a person uses an infrared camera to visually inspect components for leaks. Any component that the operator can see a plume emanating from is identified as a leaker and tagged for repair.

The detection function $\beta_{F,MIR}$ is based on the signal that we expect a plume to produce in an IR camera. An IR camera does not measure CH_4 concentration directly as some suggest [12]. The signal in a pixel of the IR camera depends on the integral of the concentration of CH_4 along the path imaged by that pixel, as shown in equation 3.22 [3]. α is a proportionality factor and Λ is the path imaged by the pixel. Further discussion of Equation 3.22 is provided in Appendix A.4.

$$Signal = \alpha \int_{\Lambda} \Phi(x(s), y(s), z(s))ds \tag{3.22}$$

FEAST simulates an IR camera by generating a grid of pixels that occupy the field of view (FOV) of the camera. The FOV is defined by two angles θ_1 and θ_2 that are input variables to FEAST. In order to simulate an image, FEAST calculates the concentration path length associated with each pixel in an image. The result is shown in Figure 3.11.

In order to define detection criteria, FEAST relies on an estimate of the minimum concentration path length that will produce a detectable signal in a pixel (Γ_{min}). The value of Γ_{min} was studied by Benson et al. [3]. In experiments, Benson measured the signal in a pixel due to a range of concentration pathlengths, and then extrapolated to find the concentration pathlength at which the signal to noise ratio would be one. Benson concluded that a concentration pathlength of 0.2 g-m/m^3 would produce a signal to noise ratio of one. In the default settings, we use two times Benson's estimate as the lower bound on the required concentration pathlength to produce a signal in a pixel.

The concentration path length associated with a pixel depends not only on the plume, but also on the position and orientation of the camera. The strongest signal is achieved by placing the camera directly in the plume and looking along its axis toward the source. However, it may be unrealistic to assume that an IR Camera surveyor would actually manage to find that location. As a metric for which leaks will be found, we consider an image of the top of a plume from a distance of 0.5 meters, see figure 3.11. We numerically integrate the concentration profile produced by the Gaussian plume model to find a concentration pathlength for an array of pixels and then compute the fraction of pixels (F_{PD}) that are above Γ_{min}. The detection function $\beta_{F,MIR}$ is defined based on F_{PD} and a minimum detection criteria, $F_{PD,min}$:

$$\beta_{F,MIR} = \begin{cases} 1 & \text{if } F_{PD} > F_{PD,min} \\ 0 & \text{Otherwise} \end{cases} \tag{3.23}$$

The operating cost of the MIR method is calculated in exactly the same way as the operating cost of the FID survey method, albeit with different values for

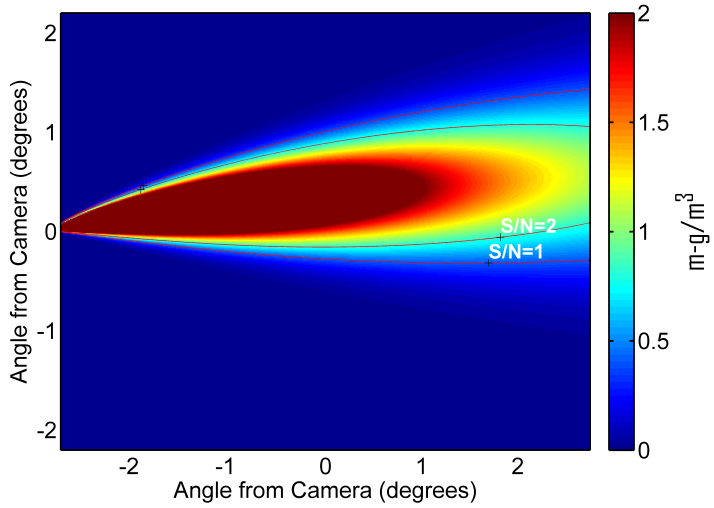

Figure 3.11: Example of an image used to determine whether a leak will be detectable or not. Color indicates the concentration-pathlength associated with each pixel. Pink lines indicate the range of pixels with a signal to noise ratio greater than one and two.

the variables. The survey cost equation (Equation 3.15) and the survey time estimate from Equation 3.19 are used. The capital cost of the MIR method is estimated as the capital cost of purchasing a Flir GasFind 320 IR camera and a truck for the operator to drive from site to site (see Table 3.6).

Table 3.6: Manual IR variables

Symbol	Name	FEAST Variable*	Units	Default Values	Sources
C_{truck} †	Truck capital	-	$	30000	-
C_{camera} †	Camera capital	-	$	90000	-
C_{Cap}	Capital cost	Capital0	$	120000	-
C_m	Maintenance cost	Maintenance0	$/year	12000	-
R_S	Survey speed	processStruct.surveySpeed	components/hour	500	[17]
Γ_{min}	Min. detectable concentration pathlength	minDetection	m-g/m^3	0.4	[3]
$F_{PD,min}$	Min. fraction of pixels above Γ_{min} for detection	minPixelCount	%	10	-
T_{SI}	Survey interval	processStruct.surveyInterval	d	100	-
T_{SU}	Setup time	processStruct.setupTime	hours	0.5	-
v_{LD}	Driving speed	processStruct.driveSpeed	m/s	15	-
T_λ	Lifetime	camStruct.lifetime	d	3650	-
θ_1	FOV angle 1	FOV1	radians	$24\pi/180$	[19]
θ_2	FOV angle 2	FOV2	radians	$18\pi/180$	[19]

* All variables are defined in the file `DetectionTechnologyModules/MIR.m`
† The sensitivity range of this variable is included in the sensitivity of C_{Cap}

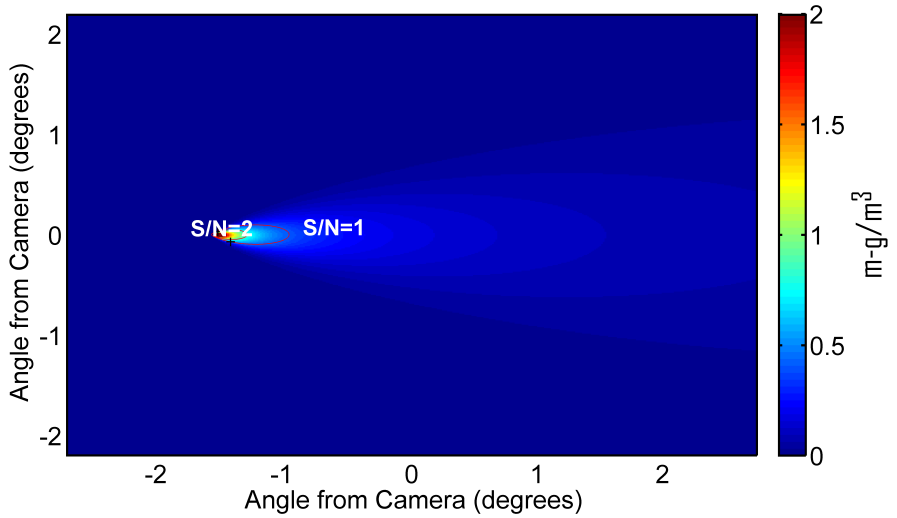

Figure 3.12: Example of an image from a drone-mounted IR camera. Color indicates the concentration-pathlength associated with each pixel. Pink lines indicate the range of pixels with a signal to noise ratio greater than one or two.

3.6.4 Automated IR camera module

The AIR program simulated here consists of a drone-mounted IR camera system that flies over the natural gas field searching for leaks. The drone is followed at all times by an operator in a truck in order to monitor its progress and take control of the drone should the need arise. Software in the camera system is assumed to be able to detect leaks without any human assistance.

The detection criteria for AIR (Equation 3.24) is identical to that for MIR (Equation 3.23), except that in calculating F_{PD} the camera is always positioned a height Z_{cam} above the ground.

$$\beta_{F,AIR} = \begin{cases} 1 & \text{if } F_{PD} > F_{PD,min} \\ 0 & \text{Otherwise} \end{cases} \tag{3.24}$$

The height of the camera Z_{cam} is much greater than the distance from the camera to the leak in the MIR module, which means that AIR cannot detect leaks as small as MIR. Figure 3.12 below shows a simulation of the same leak shown in Figure 3.11, but as it would be seen from the drone-mounted IR camera at a height of 20 meters. While in Figure 3.11 much more than 10% of the pixels are above the detection limit and the leak is classified as detected. Here less than 1 % of the pixels are illuminated and the leak is not detected. In order for the drone-mounted IR camera to detect the leak from 20 meters, the leak must have a greater flux or the atmosphere must have a lower wind speed (or both).

The capital costs for AIR include many components. In order to improve detection accuracy, we assume a hypothetical system with two IR cameras will be used with scientific-grade narrow bandwidth filters to allow differencing

between images taken at the resonant frequency of CH_4 to images taken at frequencies slightly above or below the resonant frequency. The cost of a large quadcopter drone capable of carrying both cameras is also included, as well as the cost of a truck for the operator to drive from site to site.

The operating cost is calculated based on the survey cost equation (Equation 3.15). The AIR survey time ($T_{s,AIR}$) is calculated according to Equation 3.25. The method is similar to that used to calculate the FID survey time (Equation 3.19), with two differences. The AIR program does not require any setup at each well, but does require periodic refueling stops. Therefore, FEAST accounts for the setup time by assuming that the total survey time will be greater than the flight time of the drone by a factor F_{SU} that is defined as an input variable. F_{SU} allows for the operator to land the drone periodically to change its batteries or refuel.

$$T_{S,AIR} = (T_I + T_D)\, F_{SU} N_W \tag{3.25}$$

The second difference between the FID and AIR calculations of the survey time lies in the method for computing the the inspection time (T_I), as shown in Equation 3.26. The equation calculates the amount of time for the drone to fly a grid pattern over a natural gas well based on the area surrounding the well to be surveyed (A_W), the width of the ground area imaged by the camera (W_{cam}), and the speed at which the drone flies during the survey v_S. The drone is assumed to follow the flight pattern represented in Figure 3.13, such that all of the ground around the well is flown over twice, and the area up to one half of the camera height is imaged at least once. The inspection time is estimated using Equation 3.20.

$$T_I = \frac{2A_W}{W_{cam} v_S} \tag{3.26}$$

The width that the camera images is calculated using Equation 3.27. It is assumed that the drone flies at a constant height Z_{cam}, and is pointed vertically downward.

$$W_{cam} = Z_{cam} \tan(\theta_1/2) \tag{3.27}$$

The time to drive between wells is calculated in the same way as it is in the FID and MIR modules (Equation 3.21).

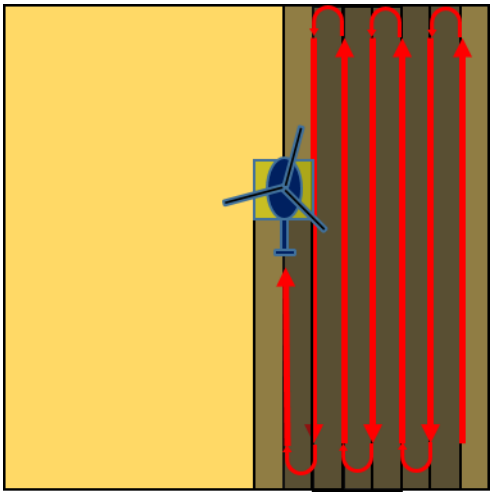

Figure 3.13: The expected flight pattern of the drone-mounted IR camera detection program. The yellow square indicates the area imaged by one frame of the IR camera.

Table 3.7: AIR variables

Symbol	Name	FEAST Variable*	Units	Default Values	Sources
C_{cam}‡	Cost per camera	-	$	50000	[17]
$C_{filters}$‡	Cost of filters	-	$	10000	-
C_{drone}‡	Cost of drone	-	$	50000	-
C_{truck}‡	Cost of truck	-	$	30000	-
$C_{software}$‡	cost of software	-	$	3000	-
$C_{capital}$‡	Total capital cost	Capital0	$	193000	-
C_m	Annual maintenance cost	Maintenance0	$/y	10	-
$F_{PD,min}$	Min. fraction of pixels above Γ_{min} for detection	minPixelCount	%	10	-
F_{SU}	Setup factor	processStruct.setupFactor	Ratio†	1.3	-
Γ_{min}	Min. detectable concentration pathlength	minDetection	m-g/m^3	0.4	[3]
T_{SI}	Survey interval	processStruct.surveyInterval	d	14	-
v_S	Survey speed	processStruct.surveySpeed	m/s	5	-
v_D	Driving speed	driveSpeed	m/s	15	-
Z_{cam}	Camera height	camHeight	m	20	-
T_λ	Lifetime	camStruct.lifetime	d	1095	-
θ_1	FOV angle 1	FOV1	radians	$5.5\pi/180$	[18]
θ_2	FOV angle 2	FOV2	radians	$4.4\pi/180$	[18]

* All variables are defined in the file `DetectionTechnologyModuls/MIR.m`.
† The setup factor is the ratio of total survey time to time when the drone is flying. This allows the drone to land for refueling or other unforeseen circumstances.
‡ The sensitivity range of this variable is included in the sensitivity of C_{Cap}

3.6.5 Distributed detector module

The DD method consists of a set of CH_4 detectors placed around each natural gas well. If the detectors sense a concentration above a particular threshold, they record a leak. Operators must then go out with a secondary detection tool to pinpoint the location of the leak and repair it. In FEAST, the DD module allows leaks to accumulate for a period of time T_{LA} [d] before repairing leaks to reduce the cost burden of sending workers out every time a leak is detected. Methods using a small number of high-cost, high-sensitivity detectors can be simulated using this module, as well as methods using a large number of low-cost, low-sensitivity detectors.

Each well has a set of detectors associated with it $k \in \mathbb{Z}_{Det}$. The position of each sensor is defined relative to the center of the well by $\vec{r}_k = (x_k, y_k, z_k)$. Finally a detection criteria Φ_{min} [g/m^3] must be defined for all of the sensors. The detection function is given in Equation 3.28, where $\Phi_i(\vec{r}_k, u(t), s(t))$ is the concentration of CH_4 due to leak i at the position $\vec{r}_k$ and atomspheric conditions $u(t), s(t)$. The value of Φ_i is calculated using the gaussian plume model.

$$\beta_{F,sniff}(Q_i, u(t), s(t)) = \begin{cases} 1 & \text{if } \max\limits_{k \in \mathbb{Z}_{Det}} \Phi_i(\vec{r}_k, u(t), s(t)) > \Phi_{min} \\ 0 & \text{Otherwise} \end{cases} \tag{3.28}$$

Unlike the survey based modules, the detector module searches for leaks at every time step.

Calculating the capital and operating costs of the DD method is different from calulating these costs for survey based methods. The capital cost not only includes the cost of purchasing the detectors, but also the cost of purchasing a GasFind IR camera to locate leaks once they have been detected. The total capital cost $C_{0,DD}$ is calculated according to Equation 3.29, where C_{cam} is the capital cost of the handheld IR camera, C_{DD} is the capital cost per detector, and $N_{DD/W}$ is the number of detectors per well.

$$C_{0,DD} = C_{cam} + C_{DD} N_{DD/W} N_W \tag{3.29}$$

Although the DD program modeled here is not dependent on periodic leak surveys, a periodic search to locate leaks that have been identified must be carried out. The method used to calculate the cost of locating leaks periodically is very similar to the survey cost equation (Equation 3.15), with three differences as shown in Equation 3.30. $\mathbb{Z}_W$ is the set of all wells.

$$C_{O,DD}(t) = C_{m,DD}\delta t + \frac{C_P T_S}{N_W \tilde{\beta}(t, T_{LI}) N_{DD/W}} \sum_{j \in \mathbb{Z}_W} \beta_L(j) \tag{3.30}$$

The differences between the above equation and Equation 3.15 are:

- Only wells with an identified leak are searched. This is enforced by

$$\beta_L(j) = \begin{cases} 1 & \text{if a leak has been detected in well } j \\ 0 & \text{otherwise} \end{cases}$$

- The search time is reduced by a factor of $1/N_{DD/W}$. This factor assumes that analysis of the signal from the detectors throughout the natural gas well will make it possible to identify the detector that is closest to the leak. That information would allow an IR camera operator to search only a fraction of the total number of components equal to $1/N_{DD/W}$.
- The survey interval T_{SI} is replaced by the time interval between location campaigns T_{LI}

Table 3.8: Distributed detector variables

Symbol	Name	FEAST Variable*	Units	Default Values	Sources
C_{cam}	Camera cost	-	$	90,000	-
$C_{detector}$	Cost per detector	SnifferCost	$	500	-
C_m	Maintenance cost	Maintenance0	$/y	21000	-
C_{truck}	Cost of truck	-	$	30000	-
C_P	Personnel cost	financeStruct.Labor	$/hour	100	-
$N_{DD/W}$	detectors per well	snifferStruct.N	-	4	-
$\vec{r}_k$ †	detector position	snifferStruct.x,y,z	m		-
R_S	IR camera speed	processStruct.surveySpeed	components/hour	500	[17]
T_{LI}	Location interval	processStruct.repairInterval	d	50	-
T_{setup}	Setup time	processStruct.setupTime	hours	0.5	-
v_D	Driving speed	processStruct.driveSpeed	m/s	15	-
Φ_{min}	Min. concentration	snifferStruct.Sensitivity	g/m^3	10^{-2}	-

* All variables are defined in the file `DetectionTechnologyModules/DD.m`.
† The position of each detector is defined by its x, y, and z coordinates (see Table 3.9). In FEAST, these coordinates are stored in the x, y, and z fields of snifferStruct.

Table 3.9: Detector positions

	x	y	z
Detector 1	-3.5	-10	2
Detector 2	0	-10	3
Detector 3	3.5	-10	4
Detector 4	0	10	4

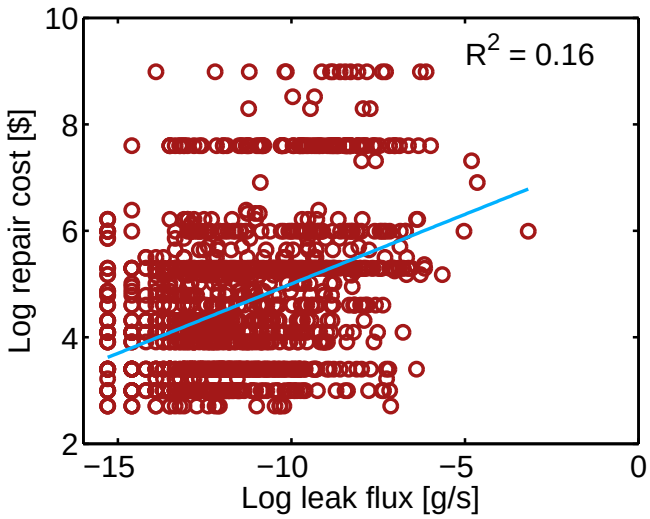

Figure 3.14: Scatter plot of repair costs versus leak size with linear fit to the log of the data. The adjusted R-squared value is shown in the top right corner.

3.7 Repair costs

The repair cost for each leak is drawn randomly from a published distribution of leak costs. Previous studies have published repair cost statistics based on the type of component to be repaired, but the difference in repair cost between component types is small in comparison to the range of repair costs that are possible for each component type [42]. FEAST does not differentiate between component types. The only publicly available dataset to estimate leak costs for individual leaks was released in the 2006 study by Fernandez [17]. The data are for 1160 leaks in natural gas plants, 609 leaks in compressor facilities, and 132 leaks from natural gas wells. Although the current version of FEAST is intended to model gas wells rather than gas plants, FEAST uses data from Fernandez under default settings because it is the most extensive leak repair cost dataset.

The data published by Fernandez show that the cost to repair leaks is largely independent of the size of the leaks, as shown in Figure 3.14. Figure 3.14 displays the data on a log-log scale and shows a linear fit to the logarithm of the data. The horizontal lines in the data are a result of the cost estimation process used by Fernandez. It is apparent from Figure 3.14 that the cost of repairing a leak of any size can span several orders of magnitude. The adjusted R-squared value of 0.16 shows that there is extremely little, if any, correlation between the leak size and repair cost.

Since the repair cost is independent of the leak size, we are able to use repair cost data published by Carbon Limits in a sensitivity case. The data from Carbon Limits do not relate repair costs with any particular leak. Instead, Carbon Limits reports the median, mean, maximum and minimum repair cost for four classes of components [42]. In order to use the information we develop

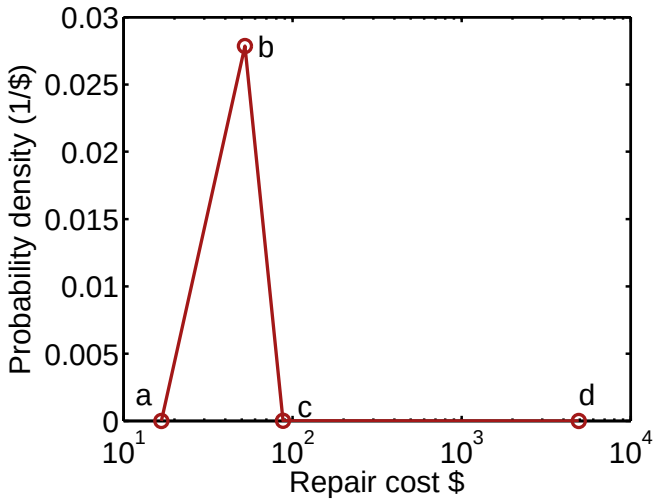

Figure 3.15: Probability density function computed using data from Carbon Limits [42]. Point c is has a value of 5×10^{-6} $\$^{-1}$.

a probability distribution based on the statistics provided by Carbon Limits.

The probability density function $P(C_R)$ that we developed is shown in Figure 3.15. In order to produce $P(C_R)$, we took the average of each statistic over the four types of components given by Carbon Limits, weighted by the number of each type of component in the Carbon Limits database. We used the mean, median and extrema provided by Carbon Limits to construct a probability distribution that consists of straight lines connecting four points (the distribution is nearly triangular, with an extended tail to allow for extremely rare high cost repairs). Points a and d are located at the weighted mean of the minimum and maximum repair costs, respectively, and $P(a) = P(d) = 0$. Point b is located at the weighted average of the median repair costs. Point c was calculated using Equation 3.31.

$$c = a + 2(b - a) \tag{3.31}$$

The height of points b and c were determined by the constraints of Equations 3.32 and 3.33.

$$\int_a^d P(C_R) dC_R = 1 \tag{3.32}$$

Equation 3.32 states that there is a probability of 1 that every repair cost is between a and d. Equation 3.33 requires the mean of the distribution to be equal to the weighted average of the mean repair costs reported by Carbon Limits (μ_{CL}):

$$\mu_{CL} = \int_a^d P(C_R) C_R dC_R \tag{3.33}$$

As Figure 3.15 shows, point $P(C_R = c)$ must be very near zero in order to match the mean given in [42]. High repair costs are possible, but extremely

rare. FEAST then randomly draws repair costs from the constructed distribution to simulate leak repair costs using the Carbon Limits report.

3.8 Financial calculations

Once FEAST completes a simulation, the net present value (NPV) of each LDAR program is calculated according to Equation 3.34. Here, t_k is the time in the simulation after $k-1$ time steps [d], $V_L(t_k)$ accounts for the value of the leakage saved by the LDAR program (see Equation 3.35), C_Σ is the total cost as in Equation 3.12, and R_{RD} is the real discount rate [%/y][3].

$$NPV = \sum_{k \in \mathbb{Z}_t} \left(V_L(t_k) - C_\Sigma(t_k)\right) \left(\frac{1}{1+R_{RD}}\right)^{t_k} \tag{3.34}$$

We do not attempt to forecast an inflation rate for the value of natural gas or other parameters. Rather, we include a real discount rate and explore the sensitivity of the results to natural gas prices and discount rates.

We focus on the marginal costs and benefits of the LDAR programs in comparison to the null scenario. The V_L of each LDAR program is calculated based on the assumed natural gas price (C_g), the gas saved in the LDAR program in comparison to the null scenario, and the length of a time step in the simulation (δt), as shown in Equation 3.35:

$$V_L(t_k) = C_g \left(L_N(t_k) - L_T(t_k)\right) \delta t \tag{3.35}$$

Similarly, the costs of the program are computed as the increase in costs over the null scenario. The default values assumed for C_g and R_{RD} are shown in Table 3.10. We refer to the NPV calculated according to Equations 3.34 and 3.35 as the "null NPV."

Alternatively, the NPV of the LDAR programs can be computed by comparing the leakage under each program to an imaginary scenario in which no leaks are repaired. We refer to the NPV calculated in this way as the "no repair NPV." In the no repair scenario, the leakage grows indefinitely throughout the simulation.

[3]The units for time given in brackets are the units that must be used to enter each variable into FEAST. FEAST internally converts the units of time to make Equation 3.34 consistent.

Table 3.10: NPV variables

Symbol	Name	FEAST Variable*	Units	Default Value	Sources
C_g	Gas price	financeStruct.gasPrice	$/mcf†	5	[46]
R_{RD}	Real discount rate	financeStruct.discountRate	% per y	8	-

* All variables are defined in the file `FinancialAssumptions.m`.
† Although FEAST defines the natural gas price in units of $/g, the values given here are in the more common units of $/mcf.

3.9 Producing results

The previous sections have described all of the components of FEAST as separate entities. FEAST code integrates the gas field simulation, atmosphere simulation and the technology modules so that every scenario can be meaningfully compared. The work flow is outlined below (see Figure 3.16):

- The natural gas field is initialized
- Atmosphere variables are initialized
- Technologies are initialized
- At each time step several operations are performed:
  - New leaks are added to each technology's leak set
  - Leaks are identified and repaired by each technology
  - The resulting set of leaks for each technology is summed and saved as the total leakage for that time step.
- Cash flows and leakage rates are saved to output files
- NPV is calculated

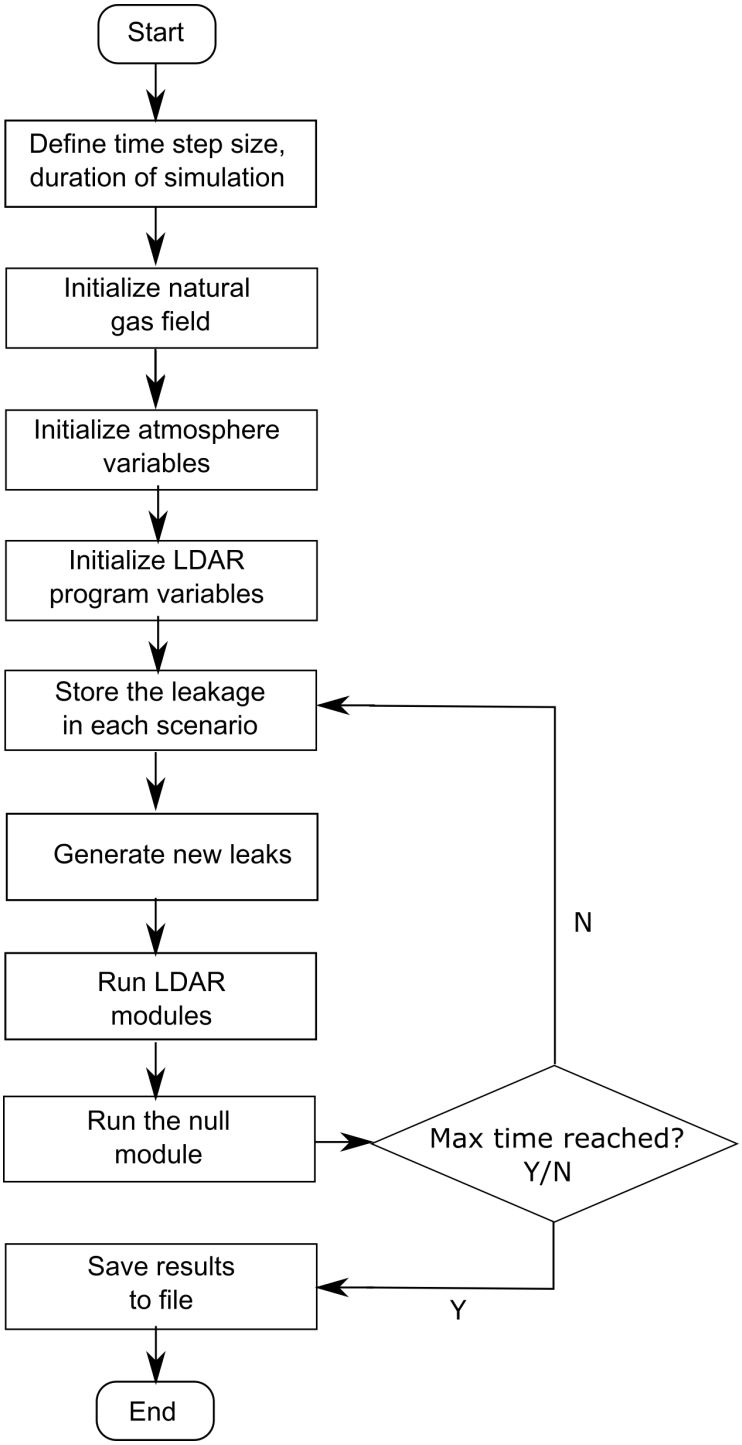

Figure 3.16: FEAST work flow diagram

4 User Guide

4.1 User interface

FEAST is stored in a directory with five subfolders. In order to run FEAST, all of these folders must be added to the current Matlab set path. Once that has been done, simply calling the function `FieldSimulation.m` will run FEAST with the default settings. Settings may be changed by editing the initialization functions called by `FieldSimulation.m`. Settings that control the atmosphere, properties of the natural gas field, and financial assumptions are stored in `InitializationFunctions`. Technology parameters are set by initialization functions in the folder `DetectionTechnologyModules`. The number and length of timesteps in the simulation are set within `FieldSimulation.m` directly.

`FieldSimulation.m` automatically generates results and saves them to the folder `Results` in the FEAST directory. If the folder does not exist, FEAST creates it automatically. Each time a new result file is generated, FEAST stores it in the folder `Results` and gives the file a unique name.

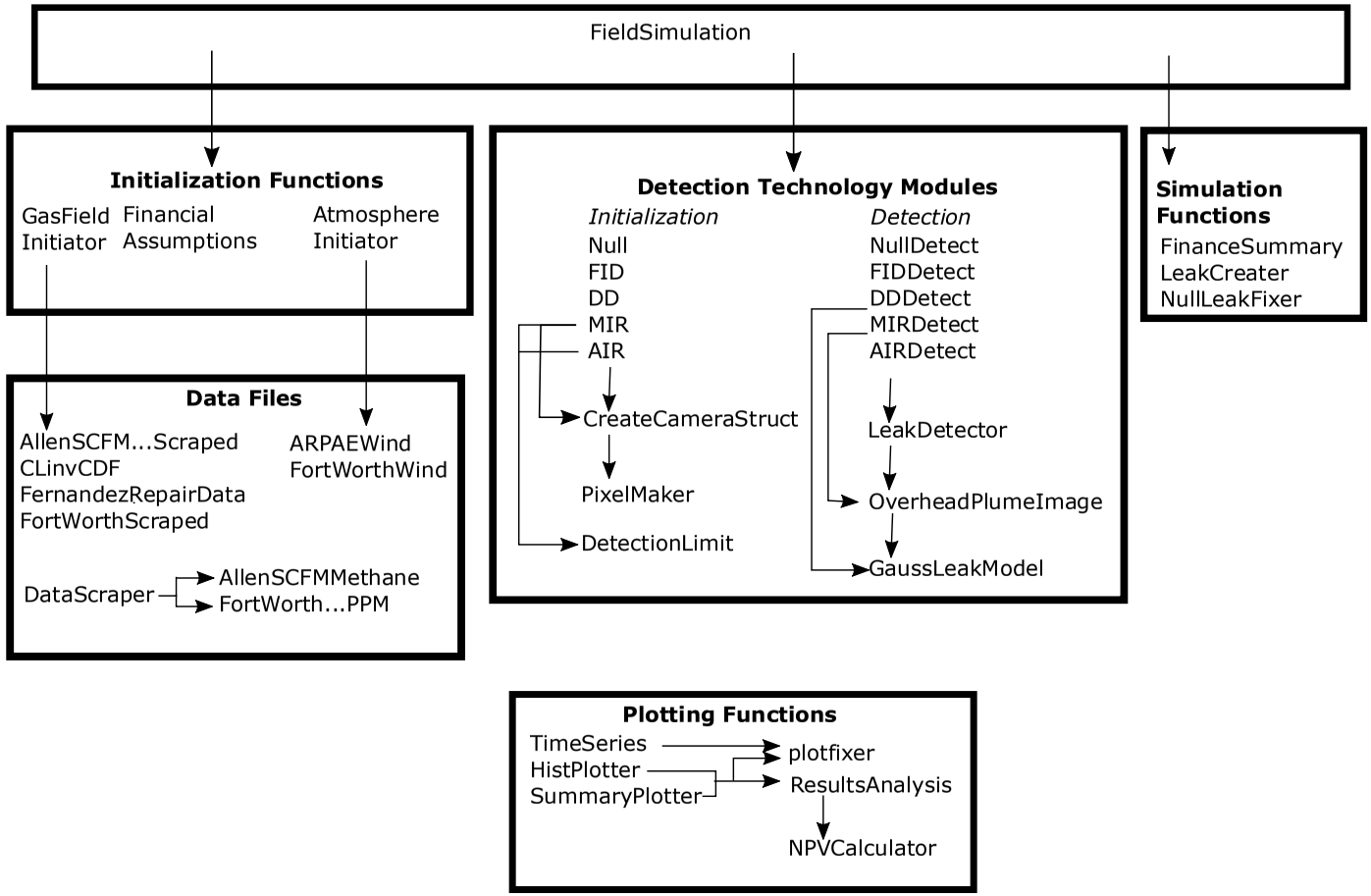

Figure 4.1: Map of files included in feast. Bold text indicates directories. Arrows point from a parent function to a subfunction called by the parent function.

FEAST is divided into five types of files: data files, intitialization functions, detection technology modules, simulation functions and plotting functions. These files are sorted into folders within FEAST, as shown in Figure 4.1.

4.2 Data Files

Data files contain basic data that are used by FEAST. Some published data files were cleaned before being used in FEAST. In those cases, both the raw data files and the cleaned data files are included, as well as a script that was used to clean the raw data files.

4.2.1 Leakage Data

There are two default leakage data sets included in FEAST: one from the study by Allen et al [1], and the second from the Fort Worth study [12]. The raw data from Allen et al are stored in the file `AllenSCFMMethane.mat`. The data are stored in a vector of leak sizes, in units of standard cubic feet per minute (scfm). The data are copied from the "mean methane (scfm)" data released in association with the paper by Allen et al. [1]. The data set was cleaned by removing leaks with a reported leakage of zero scfm, as shown in the script `DataScraper.m`. The cleaned data set is stored in `AllenSCFMethaneScraped.mat` for use by FEAST.

The raw Fort Worth data are stored in `FortWorthPercentCFMCameraBoolPPM.mat`. The data are from the `EmissionsCalculations.xlsx` workbook available from the city of Fort Worth. The data are stored in a 2127 by 3 array. Each row in the array is associated with one leak. The first column stores the total gas flux measured for the leak in units of cubic feet per minute (cfm), copied from the *%CFM* column in `EmissionsCalculations.xlsx`. The second column is 0 for leaks that were found using an IR camera, and 1 for leaks found using a FID. The third column records the concentration observed during the high-flow sampler flux measurement (ppmv).

Leak records were scraped from the raw data for two reasons. Leaks with a recorded flux of 0 cfm were eliminated. Leaks with a reported concentration of 0.001 ppm were also removed from the dataset. 0.001 was used as a default value for leaks that had a methane concentration in the hi-flow sampler exhaust that was too small to measure during the study. The resulting leak size was then calculated based on the assumed methane concentration, rather than measured directly. We chose to omit leaks that were too small to measure directly from the cleaned data set.

The cleaned Fort Worth data are stored in `FortWorthScraped.mat`. `FortWorthScraped.mat` is a data struct with seven fields. The leaks identified by an IR camera are stored in the field "IR", while leaks identified by the FID are stored in the field "FID" (g/s). The IR and FID fields are combined in the field "allLeaks" (g/s). The concentrations measured in the high-flow sampler exhaust are recorded in the "concentrations" field. A few populations

statistics from the data are stored in the remaining three fields: the average number of leaks per well identified with the IR camera are stored in "camLeakRate", the average number of leaks per well identified using both the IR camera and the FID is stored in "genLeakRate", and the average leak size is stored in "avgLeak" (g/s).

4.2.2 Leak repair costs

Two sources for leak repair costs are stored in FEAST: data published by Fernandez [17] and data published by Carbon Limits [42]. Leak repair costs estimated by Fernandez are stored in `FernandezRepairCost.mat`. `FernandezRepairCost.mat` is simply a vector of repair costs ($).

The statistics published by Carbon Limits are stored and used to develop an inverse CDF in `CLinvCDF.m` (see Section 3.7). `CLinvCDF.m` is a function that takes a number between 0 and 1 and returns a repair cost according to the computed inverse CDF. Passing a set of random numbers chosen from a uniform distribution between 0 and 1 to `CLinvCDF.m` will return a set of repair costs that follows the repair cost distribution calculated from the Carbon Limits data.

4.2.3 Wind data

Two wind data sets are included. `ARPAEWind.csv` has hourly wind data provided by the ARPA-E MONITOR program [33]. The data are in three columns: the date, the hour in the year, and the average hourly wind speed at 2 meters (m/s). Wind data from Fort Worth are stored in `FortWorthWindData.csv` [36]. The data are in three columns: the average daily windspeed (mph), the maximum 2 minute windspeed for each day (mph) and the average wind direction (degrees from North).

4.3 Initialization functions

Initialization functions load data, define input variables, and generate random variables that will be used in FEAST simulations. There are three initialization functions: `AtmosphereInitiator.m`, `FinancialAssumptions.m`, and `GasFieldIniator.m`.

4.3.1 Atmosphere initialization

`AtmosphereInitiator.m` initiates the wind speed, wind direction and stability class variables to be used in the model. Wind speed data are available from ARPA-E [33] and from the NOAA weather station in Fort Worth [36]. Changing between wind speed data sets requires changing the definition of the variable windSpeedData. The number of time steps in the simulation must be passed to `AtmosphereInitiator.m` so that the function can compute one wind speed, direction and stability class for each time step. The

expansion coefficients used to compute σ_z and σ_y are also stored for every time step based on the computed stability class for later reference. Results of `AtmosphereInitiator.m` are passed out of the function as a struct.

4.3.2 Financial assumptions

`FinancialAssumptions.m` defines the gas price and discount rate used by FEAST for economic analyses. The function requires no inputs and returns a struct with two fields: one for the gas price and one for the discount rate. The gas price and discount rate are set within the function.

4.3.3 Gas field initialization

`GasFieldInitiator.m` defines the natural gas field to be simulated. The function is divided into three sections (User Inputs, Leak Distribution, Final Output). In the first section, user defined input variables are set. The second section loads a leak size distribution and generates an initial set of leaks. The user may direct the function to load any leak size distribution here. Finally, all data entered and loaded in the function is stored in a struct to be passed out of `GasFieldIniator.m` in the third section.

4.4 Program modules

Program modules may be written by the user to model any LDAR program. In order to integrate into FEAST smoothly, every program module must have at least two functions: one that initializes the program (known as a technology initialization function or TIF), and one that determines whether or not a leak can be identified by the detection technology (known as a detection function). TIFs and detection functions each have a set of inputs and outputs that must be consistent across all technology modules. This allows each module to be invoked in a simulation with a simple *for* loop. The form of the function that maps the input arguments to the outputs is completely up to the user.

4.4.1 Technology initialization functions

TIFs may use data stored in the struct GasField (produced by running `GasFieldIniator.m`), and timeStruct (produced by `FieldSimulation.m`). In addition to accepting these two input arguments, each technology module should also allow for optional input arguments that can be used to set values in the module by command. A call to the FID initializatoin function is shown below as an example of the suggested format:

$$[\mathsf{FIDStr}, \mathsf{financeStr}, \mathsf{processStr}] = \texttt{FID}(\mathsf{GasField}, \mathsf{timeStr}, \text{'lifetimeFactor'}, 1.5) \tag{4.1}$$

GasField and timeStruct contain fields that can be used by the function to compute technology costs as a function of time. The string 'lifetimeFactor' is an optional input argument that specifies a property of the technology that will be changed from the default value, and finally the number 1.5 sets the value of that property.

Each TIF returns three structs. The first struct is referred to as the "technology struct." It contains the key performance properties of the detection technology, including the expected lifetime and the necessary information to define a detection criteria. The second struct is called the "finance struct." It contains the capital cost, maintenance cost, and other financial information about the LDAR program. The final struct is called the "process struct." It contains information about how the detection technology will be applied in the LDAR program. For example, if a technology is applied through periodic surveys, the time interval between surveys is stored in the process struct.

Within the guidelines presented above, each program module may define different fields in the reurned structs. For example, in the case of a distributed detector based LDAR program the location and minimum detection limit of each detector must be defined, where as an IR camera based program requires the minimum detectable concentration path length, the required fraction of pixels above that detection limit, and information about the position of the camera relative to leaks. The flexibility of the returned structs to contain unique fields allows for new and different LDAR programs to be incorporated into FEAST.

4.4.2 Detection functions

Detection functions take information about the detection technology, the existing leaks, repair processes, the simulation time and the atmospheric conditions to determine which leaks will be repaired and to calculate the associated costs. All detection functions must accept the following variables: TechStruct, financeStruct, processStruct, leakStruct, leaksFound, repairData, timeStruct, k, and atmosphereStruct. TechStruct, financeStruct, and processStruct are the three outputs from TIFs discussed in Subsection 4.4.1. leakStruct defines all of the existing leaks, including their size and location. leaksFound is a vector that stores all of the leaks found by each LDAR program module. repairData stores the repair cost distribution. timeStruct stores the time data that defines the simulation, including the length of the simulation and the length of each time step. k is an integer that increments by one at each time step. Finally, atmosphereStruct is the struct created by `AtmosphereInitiator.m` (recall subsection 4.3.1).

Detection functions return four variables: leaksFound, leakStruct, repairCost and financeStruct. Every detection function must update leaksFound with the new list of detected leaks and remove detected leaks from leakStruct. Detection functions must also calculate the repair cost associated with each leak and return the total repair cost at that time stop in the scalar repairCost. The finance struct may or may not be updated by a detection function: if the cost of find-

ing leaks is completely deterministic (as in the basic FID module), the finance struct may be completely calculated in the TIF. However, the some LDAR programs may have finding costs that depend on the number of leaks (the DD module simulates such a program), in which case the finance struct must be updated by the detection function. The finance struct is passed in and out of every detection function to maintain a consistent syntax for each function, regardless of whether or not a particular function affects the finance sruct.

4.4.3 Module implementation

Here we give a practical introduction to the four program modules described theoretically in Section 3.6.

The FID module is the most simple. The initialization function calculates the capital, maintenance, and finding costs of the program a priori. The detection function (`FIDDetect.m`) is called at every time step. If the simulation time is within one time step of a multiple of the survey interval, it adds all of the existing leaks to the list of found leaks. A repair cost is randomly drawn from the repair cost distribution for every leak, and the sum is stored to repairCost. Finally, every field in leakStruct is set to an empty array.

The AIR and MIR modules overlap. Both initialization functions call a subfunction `CreateCameraStruct.m` to create the technology struct for the IR cameras. The initialization functions also call a second subfunction `DetectionLimit.m` that computes the minimum flux that the technology can detect. `DetectionLimit.m` saves time during the simulation because it allows the detection function to immediately tag many leaks as too small to detect without running the full detection function. `DetectionLimit.m` takes the struct gasField and camStruct as inputs. In addition, it accepts an optional argument that defines how the camera's position is defined. In the MIR module, the camera is always assumed to be a set distance above the leak, while in the AIR module the camera is always assumed to be a set distance above the ground. The function chooses the best conditions for identifying a leak that are possible given the gasField and camStruct inputs (eg. the most stable atmosphere and a leak location that is as close as possible to the AIR flight pattern) and calculates the minimum flux that can be detected under those conditions. The resulting value is passed back to the initialization function and stored in the technology struct. Like the FID module, the capital, maintenance and finding costs are all calculated in the initialization functions.

The AIR and MIR detection functions are very similar to the FID detection function. If the simulation time is within on time step of a multiple of the survey interval, both functions identify the leaks that can be detected, update the variables leaksFound, leakStruct, and repairCost, and return the finance struct unchanged.

The AIR detection function determines which leaks are found by calling a subfunction `LeakDetector.m`. The inpus to `LeakDetector.m` are the camera technology struct, an array of leak structs, and the atmosphere struct at the

current time. Optional arguments may also be passed to `LeakDetector.m` to define the detection limit for the technology, or to specifiy how the camera position is defined (with respect to a leak or with respect to the ground).

`LeakDetector.m` generates images of each leak with the camera positioned directly downwind of the leak. In the first iteration, the camera is positioned such that the leak source is just at the upwind edge of the frame. If the leak is detected in that image, the function records the leak as found and moves onto the next leak. If the leak is not detected, the camera is moved one half of a frame width farther downwind of the leak and a new image is simulated. The function continues to iterate farther downwind until the signal begins to decrease or the leak is found.

The signal from simulated images is calculated by a second subfunction: `OverheadPlumeImage.m`. `OverheadPlumeImage.m` implements the rapid concentration pathlength integration method described in Subsection A.4.2. It takes the technology struct as an input, as well as the flux, position and buoyancy flux parameter (raised to the one-third power) of the leak; and the current wind speed and atmospheric stability class. It returns two variables: Image (an array that stores the concentration pathlength associated with each pixel), and pixelCount (the number of pixels with a detectable signal). Within `LeakDetector.m`, Image is suppressed and only pixelCount is returned. In order to calculate the concentration pathlength, `OverheadPlumeImage.m` calls a final subfunction: the Gaussian plume model stored in `GaussLeakModel.m` (see subsection 4.4.4).

The MIR module differs from the AIR module because only one image is considered for each leak. `LeakDetector.m` is not necessary as a result, and the module calls `OverheadPlumeImage.m` directly. The MIR module further differs from the AIR module because the camera position is defined with respect to the position of the leak. As a result, the signal from a leak at a particular time step exists in a one dimensional space; the only difference in the signal from different leaks at a particular time step in the MIR module is due to their size. The MIR module takes advantage of this by sorting the leaks before calling `OverheadPlumeImage.m`. The leaks are analyzed in descending order until one is not detected. All smaller leaks are not detected by the LDAR program under the atmospheric conditions of that time step.

The DD Detection function (`DDDetect.m`) is unique because detection and repair occur at different times. At every time step, the concentration at the position of every detector is calculated for every leak. The concentration is calculated using `GaussLeakModel.m`. `DDDetect.m` first rotates the coordinates of the detectors in order to match the coordinate system assumed by `GaussLeakModel.m`. `GaussLeakModel.m` defines `x`, `y` and `z` such that `x` increases downwind of the leak, while the positions of the detectors are defined such that `x` increases toward the East. The position of the detectors passed to `GaussLeakModel.m` are rotated based on the direction of the wind.

If the concentration at the location of a detector is above the detection limit, the leak is marked as found and is not checked at future time steps. Once per repair interval, all of the found leaks are repaired and removed from the array

of existing leaks.

4.4.4 Gauss leak model

`GaussLeakModel.m` implements the leak model described in Section 3.5. The plume model requires the following inputs: the position at which to calculate the concentration (x, y, z), the leak size (Q), the leak position (H0, x0, y0), the windspeed u, the plume buoyancy factor raised to the one-third power (Fonethird) and the plume dispersion coefficients (Ry, Rz). The concentration at multiple locations may by calculated simultaneously by defining x, y, and z as arrays. Similarly, the function can calculate the cumulative concentration at a point due to multiple leaks by passing in vectors for the leak variables (Q, H0, x0, and y0). The atmospheric conditions, however, must be the same at every position and for every leak. Given these inputs, `GaussLeakModel.m` calculates the concentration at the desired points according to Equation 2.1.

4.5 Field simulation

`FieldSimulation.m` is the highest level function in FEAST. It is the function that calls all of the initialization functions, loops through the LDAR program modules, and saves the generated results.

The `FieldSimulation.m` function is implemented in six sections. The *Set up* section adds all of the folders in FEAST to the set path and chooses a new seed for the random number generators. The *Simulation parameters* section defines the time steps for the simulation and the technology modules to be called. This is the only section in `FieldSimulation.m` where parameter values are entered directly. The *Initialize variables* section loads the GasField struct, the atmospheric conditions for every timestep in the simulation, the leaks to be generated at each time step, and the financial assumptions. If no arguments are passed into `FieldSimulation.m`, the initialization functions introduced in Subsection 4.3 generate the required structs. Up to three optional input arguments can be given to FEAST: a GasField struct, a series of leak structs, or a series of atmosphere structs. When provided, these inputs replace the variables generated randomly by the initialization functions. Providing any of the optional inputs to FEAST eliminates some of the stochastic variability in FEAST allowing for more direct comparison between simulations.

The fourth section, *Technology initialization* calls each TIF. In addition, it initializes several variables that store data that can be used to evaluate the program modules after the simulation. leakStruct stores all of the leaks that currently exist in the gas field under each program module. It is updated at every time step throughout the simulation. Here, leakStruct is initiated to the initial set of leaks for each LDAR program. It includes the location, flux and buoyancy flux parameter for each leak. leakStruct also includes a field to note leaks that have been detected but not yet repaired (leakStruct.leaksDetected). Finally,

leaksFound and nullRepaired store the size of leaks found by the LDAR program and by the null process, respectively.

The fifth section, entitled *Simulation*, evaluates the leakage at every time step. First, new leaks are created either by loading data from an input struct or by calling the subfunction `LeakCreater.m`. The atmosphere struct is updated to a new set of conditions, the total leakage in the absence of any repairs is saved and the list of generated leaks is updated. Finally, `FieldSimulation.m` loops through each technology module. Each detection function is called, and the leaks found, total leakage, and repair costs are saved for each technology. The same set of new leaks are added to each technology.

Depending on the simulation settings and the computational requirements of the detection functions, the *Simulation* section may take hours to run. Once it completes, `FieldSimulation.m` moves on to the sixth and final section: *Save results*. First unnecessary variables are cleared from the workspace. The subfunction `FinanceSummary.m` is called to consolidate the financial information generated by the simulation. A 'Results' folder is created (if it does not already exist), and all of the results variables are saved to a `*.mat` file.

4.6 Simulation functions

The simulation functions are called directly by `FieldSimulation.m` to simplify the simulations. The simulation functions consist of the following files: `LeakCreater.m`, `NullLeakFixer`, and `FinanceSummary.m`. Each function is discussed further in a subsequent subsection.

4.6.1 Leak creater

`leakCreater.m` creates new leaks at every time step according to the leak creation rate defined in `GasField.m`. The leak sizes are chosen from the distribution defined in `GasField.m`, and the leak buoyancy flux parameters are calculated based on the leak sizes. The leak locations are chosen from a uniform random distribution. All of the leak parameters are saved to a struct and returned by the function.

4.6.2 Null leak fixer

`NullLeakFixer.m` repairs leaks at every time step for each program module according to the rate defined in `GasField.m`. It takes the number of existing leaks, the leak repair rate and the length of a time step, and returns a set of integers that define the leaks repaired by the null process and the leaks that remain.

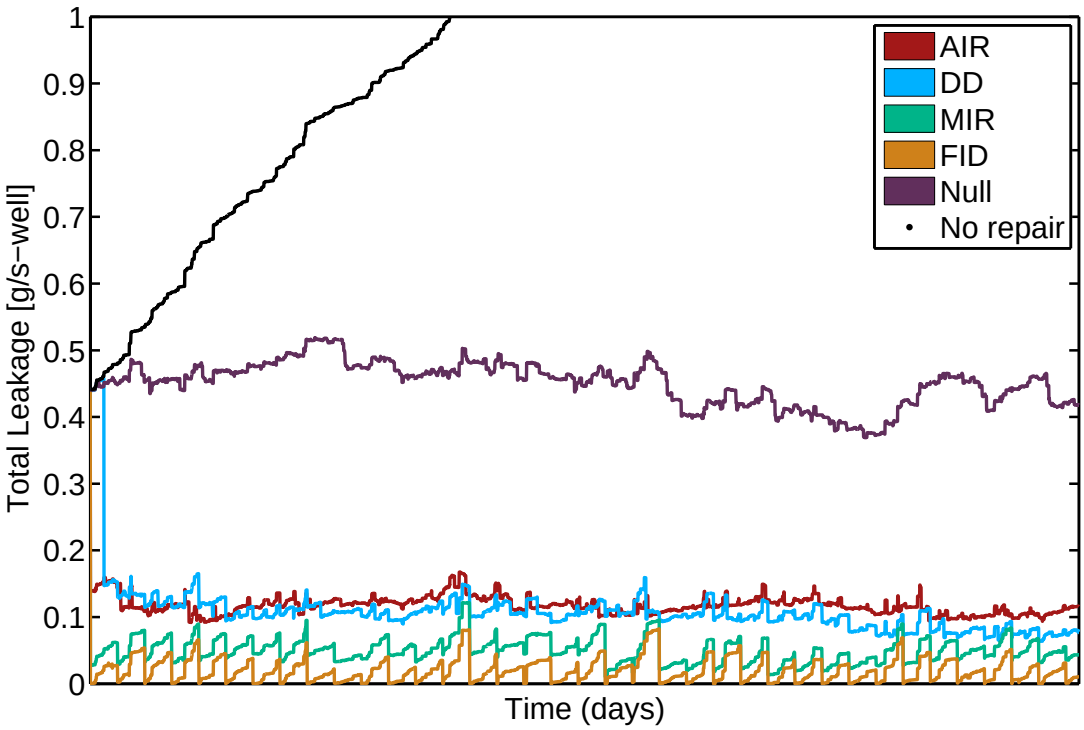

Figure 4.2: Natural gas leakage in fields with different LDAR programs applied.

4.6.3 Finance summary

`FinanceSummary.m` takes financeStruct as an input and separates data stored in the struct's maintenance, Capital and findCost fields for each LDAR program module into three separate arrays. The function does not add any new data to the results file, but allows the results to be interpreted more easily.

4.7 Plotting functions

Three basic plotting functions are included in FEAST: `TimeSeries.m`, `HistPlotter.m`, and `SummaryPlotter.m`. In addition to these three files, three helper functions are included that aid plot formatting and post-processing of simulation data.

4.7.1 Time series

`TimeSeries.m` produces a time series of the leakage under each LDAR program in the simulation results file. It simply takes the path to a results file as an input, and plots the array Leakage stored in that file. The resulting plot is formatted by `plotfixer.m`. An example is shown in Figure 4.2.

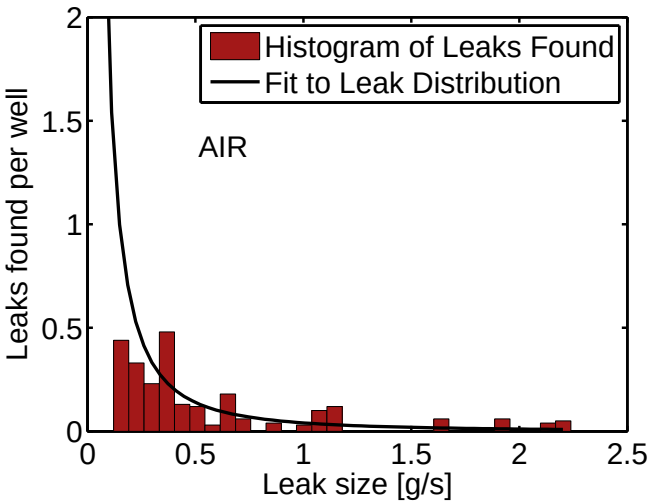

Figure 4.3: Leaks found in a ten year simulation by the AIR method under default settings.

4.7.2 Histogram plotter

`HistPlotter.m` creates a histogram of the leaks found by each LDAR program in a set of simulations. `HistPlotter.m` takes a directory of one or more results files as its input. It calls the subfunction `ResultsAnalysis` to compile the list of leaks found from every file in the directory. The first file in the directory is loaded for reference by the function.

`HistPlotter.m` generates a lognormal fit to the leak data used by the first file in the input directory. The function then creates a histogram of the size of leaks found by each technology including results from every file. Each histogram is plotted in its own figure, and the lognormal fit to the leak distribution is superimposed on the histogram. The histogram for LDAR programs that find a high fraction of the leaks closely follow the lognormal fit. An example is shown in Figure 4.3. Leaks found by the null process are not included in the plotted results (except in the null scenario, in which no explicit LDAR program is simulated).

4.7.3 Summary plotter

`SummaryPlotter.m` creates a stacked bar plot of the costs and benefits of each LDAR program. A directory of one or more results files must be passed into `SummaryPlotter.m`. The subfunction `ResultsAnalysis.m` compiles data from all of the files in the directory and returns a time series of the leakage in each LDAR program, and the NPV of each program in comparison to the Null module. The maintenance, capital, finding and repair costs are plotted for each LDAR program, as well as the value of the avoided leakage.

The mean total NPV of every results file in the directory is plotted as a circle. The standard error in the total NPV in the set of results files is indicated

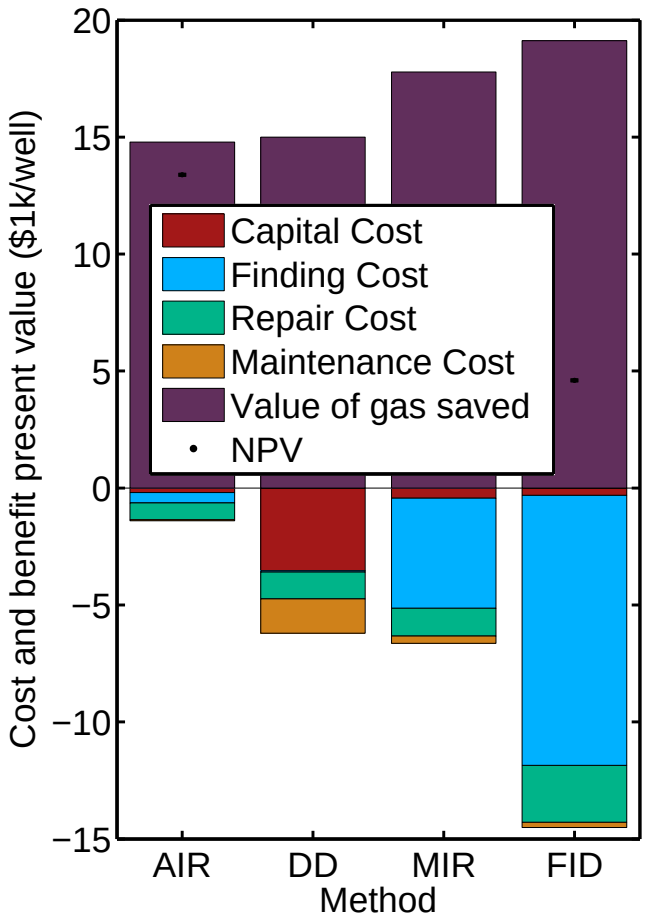

Figure 4.4: Plot generated by `SummaryPlotter.m` of the cost and benefit of various components of the LDAR programs.

by error bars about the mean. The standard error is calculated according to Equation 4.2:

$$\sigma_\mu = \frac{\sigma_{sample}}{\sqrt{n}} \tag{4.2}$$

In Equation 4.2, σ_{sample} is the sample standard deviation computed from a set of realizations of a scenario, and n is the number of realizations in the set. The standard error (σ_μ) gives a measure of the uncertainty of the mean NPV of the scenario given the sample size used.

An example summary plot is shown in Figure 4.4 below. The plot is generated based on results from a single realization.

4.7.4 Results analysis

`ResultsAnalysis.m` compiles results from multiple files in a directory. The function first loads one file from the input directory and uses it to initialize

output variables. `ResultsAnalysis.m` then loops through all of the remaining results files in the directory.

The output variables noRepairNPV and nullNPV are calculated by the sub-function `NPVCalculator.m`. noRepairNPV and nullNPV are arrays of structs. Each result file in the directory is assigned one element in the array. Each struct stores the NPV of the repair, maintenance, capital and finding costs, as well as the NPV of the gas saved by the LDAR program and the total NPV in seperate fields. The output variable leaksFound is a struct with one field for each LDAR program (including the null program). The struct stores the size of every found leak from every results file. Finally, timeSeries stores the leakage in every results file, under each LDAR program and at every time step in a three dimensional array.

4.7.5 NPV calculator

`NPVCalculator.m` takes a path to a single results file as its input and returns the null NPV and no repair NPV of each LDAR program. It calculates the NPV of the capital, finding, and maintenance costs using the discount rate stored in the results file. The NPV due to each of these costs is stored in a separate field in the output structs noRepairNPV and nullNPV. These values are the same in comparison to both the null and no repair cases. The NPV of repair costs, and NPV of the leakage saved are calculated separately in the no repair and null cases and also stored to separate fields in noRepairNPV and nullNPV. Finally the the fields in each struct are summed and the total is saved to a 'Total' field in the structs.

4.7.6 Plot formatting

Each plot generated by FEAST benefits from formatting commands stored in the `plotfixer.m` script [44]. The script adjusts the font, lines and markers in the plots to improve on the default plot settings.

4.8 User Results

Each time FEAST runs it stores 20 variables to a results file. These variables include all of the input settings used to run FEAST, the evolution of the state of the field throughout the simulated time period, and performance data for the LDAR programs. The variables are summarized in Table 4.1.

Many of the variables contain data about all of the simulated LDAR programs. The results are ordered consistently in all of these variables. The order is defined by legendString.

Table 4.1: Guide to result files

Variable name	Variable type	Description	Units
Capital	Array of numbers (N_T by $N_{\delta t}$)†	Equipment procurement costs at each timestep	$
Finding	Array of numbers (N_T by $N_{\delta t}$)	LDAR program operating costs at each timestep	$
GasField	Struct (16 fields)	Gas field settings and leak state at the end of the simulation	-
Leakage	Array of numbers (N_T by $N_{\delta t}$)	Total leakage at each time step	g/s
Maintenance	Array of numbers (N_T by $N_{\delta t}$)	Maintenance cost at each time step	$
Ntechnologies	Scalar	Number of LDAR programs simulated	-
atmStruct	Struct (5 fields, N_{δ_t} elements each)	Atmospheric conditions at each time step	-
econStruct	Struct (2 scalar fields)	Defines the assumed gas price and discount rate	-
financeStruct	Array of N_T structs	Defines all financial information for every LDAR program	-
leakList	Array (N_L)	List of all leaks generated to initialize and run the simulation	-
leakStruct	Array of N_T structs (6 fields each)	Defines the leaks currently in existence for each program	-
leaksFound	Array of N_T structs (1 field each)	List of the size of every leak found by each program	g/s
legendString	Array of strings (N_T)	The name of each LDAR program simulated	-
newLeaks	Array $N_{\delta t}$ structs (6 fields each)	Defines the leaks produced at each time step	-
nullRepaired	Array of N_{δ_t} structs (1 field each)	List of leaks repaired by the null process in each program	g/s
processStruct	Array of N_T structs	Defines details about the process followed for each LDAR program	-
repairCost	Array (N_T by $N_{\delta t}$)	The repair cost for each LDAR program at every time step	$
techStruct	Array of structs (N_T structs)	Properties of each LDAR technology	-
timeStruct	Struct (4 fields)	Defines time properties for the simulation	-
varargin	Struct	Lists optional input arguments used in the simulation	-

† N_T is the number of LDAR pograms simulated plus one to include the null program, and $N_{\delta t}$ is the number of timesteps in the simulation

Bibliography

[1] D. Allen, V. M. Torres, et al. "Measurements of methane emissions at natural gas production sites in the United States David". *Proceedings of the National Academy of Sciences* 110.44 (Sept. 2013), pp. 18025–18030. ISSN: 0027-8424. DOI: 10.1073/pnas.1315099110. URL: http://www.pnas.org/cgi/doi/10.1073/pnas.1315099110.

[2] R. A. Alvarez, S. W. Pacala, et al. "Greater focus needed on methane leakage from natural gas infrastructure". *Proceedings of the National Academy of Sciences* 109.15 (2012), pp. 6435–6440. ISSN: 0027-8424. DOI: 10.1073/pnas.1202407109.

[3] R. G. Benson, J. A. Panek, et al. "Direct Measurements of Minimum Detectable Vapor Concentrations Using Passive Infrared Optical Imaging Systems". *AWMA Proceedings* 1025 (2006).

[4] M. Beychock. *Fundamentals of Stack Gas Dispersion*. Fourth. Newport Beach CA, 2005.

[5] J. Bourke, K. Mire, et al. *Fugitive Hydrocarbon Emissions from Oil and Gas Production Operations*. Tech. rep. Star Environmental for the American Petroleum Institute, 1993.

[6] A. Brandt, G. A. Heath, et al. "Methane Leaks from North American Natural Gas Systems". *Science* 343 (2014), pp. 733–735.

[7] G. Briggs. "Chimney plumes in neutral and stable surroundings". *Atmospheric Environment (1967)* 6.7 (1972), pp. 507–510. ISSN: 00046981. DOI: 10.1016/0004-6981(72)90120-5.

[8] G. Briggs. "Optimum Formulas for Buoyant Plume Rise". *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences* 265.1161 (1969), pp. 197–203. ISSN: 1364-503X. DOI: 10.1098/rsta.1969.0048.

[9] G. Briggs. *Diffusion Estimation for Small Emissions*. Tech. rep. Oak Ridge, TN: NOAA, 1973.

[10] G. Briggs. "Plume Rise and Buoyancy Effects". In: *Amtospheric Science and Power Production*. Ed. by D. Randerson. Technical Information Center Office of Scientifiic and Technical Information United States Department of Energy, 1984, pp. 327–361.

[11] A. Cimorelli, S. Perry, et al. *AERMOD: Description of model formulation*. Tech. rep. Research Triangle Park, North Carolina: US Environmental Protection Agency, 2004, pp. 1–91. DOI: EPA-454/R-03-004. URL: http://scholar.google.com/scholar?hl=en\&btnG=Search\&q=intitle:AERMOD:+DESCRIPTION+OF+MODEL+FORMULATION\#0.

[12] *City of Fort Worth Natural Gas Air Quality Study*. Tech. rep. Fort Worth, TX: Eastern Research Group and Sage Environmental Consulting LP. for the City of Fort Worth, 2011. URL: http://fortworthtexas.gov/uploadedFiles/Gas_Wells/AirQualityStudy_final.pdf.

[13] Colorado Air Pollution Control Division. *Cost-benefit analysis for proposed revisions to Colorado Air Quality Control Commission Regulation Number 3 (5 CCR 1001-5) and Regulation Number 7 (5 CCR 1001-9)*. Tech. rep. 3. Colorado Air Quality Control Commission, 2014.

[14] W. Eaton. *Fugitive Hydrocarbon Emissions from Petroleum Production Operations*. API publication: American Petroleum Institute. American Petroleum Institute, 1980. URL: https://books.google.com/books?id=BjthtwAACAAJ.

[15] *Economic Analysis of Methane Emission Reduction Opportunities in the US Onshore Oil and Natural Gas Industries*. Tech. rep. ICF International for the Environmental Defense Fund, 2014. URL: https://www.edf.org/sites/default/files/methane_cost_curve_report.pdf.

[16] *Emissions Measurement Center - Method 21 - Determination of Volatile Organic Compound Leaks*. Tech. rep. US EPA, OAR, Office of Air Quality Planning and Standards (OAQPS), 2011, pp. 1151–1166.

[17] R. Fernandez. *Cost Effective Directed Inspection and Maintenance Control Opportunities at Five Gas Processing Plants and Upstream Gathering Compressor Stations and Well Sites*. Tech. rep. March. National Gas Machinery Laboratory, Clearstone Engineering Ltd. Innovative Environmental Solutions, Inc., 2006.

[18] FLIR. *FLIR A6700sc*. 2013. (Visited on 01/01/2015).

[19] FLIR. *FLIR GF 320*. 2009. URL: http://www.mjwilsongroup.co.uk/datasheets/FLIR-GF320-Series.pdf (visited on 05/18/2015).

[20] E. Fosler-Lussier. "Markov Models and Hidden Markov Models: A Brief Tutorial". *International Computer Science Institute* Ca Tr-98-041 (1998).

[21] *Gaussian Dispersion with MOS Forecast Data*. 2015. URL: https://ready.arl.noaa.gov/gaussian-bin/mossrc.pl (visited on 04/24/2014).

[22] F. A. Gifford. "Use of Routine Meteorological Observations for Estimating Atmospheric Dispersion". *Nuclear Safety* 2.4 (1961), pp. 47–51.

[23] *Google Earth: Bakken Well Spacing. 48.35 degrees N. 103.28 degrees W.* 2013. (Visited on 05/05/2015).

[24] *Google Earth: Barnett Gas Wells. 33.10 degrees N. 97.49 degrees W.* 2014. (Visited on 05/15/2015).

[25] *Google Earth: Fort Worth gas wells 32.06 degrees N 97.40 degrees W.* 2014. (Visited on 11/05/2014).

[26] R. W. Howarth, R. Santoro, et al. "Methane and the greenhouse-gas footprint of natural gas from shale formations". *Climatic Change* 106 (2011), pp. 679–690. ISSN: 01650009. DOI: 10.1007/s10584-011-0061-5.

[27] *Inventory of U . S . Greenhouse Gas Emissions and Sinks : 1990-2012*. Tech. rep. Washington, DC: US Environmental Protection Agency, 2014.

[28] A. Ishkov, G. Akopova, et al. *Understanding Methane Emissions Sources and Viable Mitigation Measures in the Natural Gas Transmission Systems : Russian and US Experience*. Tech. rep. International Gas Union Research Conference, 2011. URL: http://www.globalchange.umd.edu/data/publications/IGU_Research_Conference_2011_Paper_2011-0715-finalv2.pdf.

[29] A. Karion, C. Sweeney, et al. "Methane emissions estimate from airborne measurements over a western United States natural gas field". *Geophysical Research Letters* 40.16 (Aug. 2013), pp. 4393–4397. ISSN: 00948276. DOI: 10.1002/grl.50811. URL: http://doi.wiley.com/10.1002/grl.50811.

[30] W. Klug. "A method for determining diffusion conditions from synoptic observations". *Staub-Reinhalt. Luft* 29 (1969), pp. 14–20.

[31] J. Kuo. *Final project report estimation of methane emissions from the California Energy Commission*. Tech. rep. California Air Resources Board, 2012. DOI: 500-09-007. URL: http://www.energy.ca.gov/2014publications/CEC-500-2014-072/CEC-500-2014-072.pdf.

[32] J. Logan, G. Heath, et al. "Natural Gas and the Transformation of the U . S . Energy Sector : Electricity Natural Gas and the Transformation of the U . S . Energy Sector : Electricity". November (2012).

[33] *Methane Observation Networks with Innovative Technology to Obtain Reductions – MONITOR*. 2014. URL: http://arpa-e.energy.gov/sites/default/files/documents/files/MONITORandDELTAProjectDescriptions_Final_12.15.14.pdf (visited on 04/21/2015).

[34] M. Modest. *Radiative Heat Transfer*. 3rd. San Francisco, CA: Elsevier, 2013.

[35] G. Myhre, D. Shindell, et al. "Climate Change 2013: The Physical Science Basis. Contribution of Working Group I to the Fifth Assessment Report of the Intergovernmental Panel on Climate Change". In: *Climate Change 2013: The Physical Science Basis. Contribution of Working Group I to the Fifth Assessment Report of the Intergovernmental Panel on Climate Change*. Ed. by T. Stocker, D. Qin, et al. Cambridge, United Kingdom and New York, NY, USA, 2013. Chap. Anthropoge, pp. 659–740.

[36] *NOAA National Weather Service Forecast Office Dallas/FortWorth, TX Climate Data*. 2014. URL: http://www.srh.noaa.gov/fwd/?n=dfwclimo (visited on 03/01/2015).

[37] F Pasquill. "The estimation of the dispersion of windborne material". *The Meteorological Magazine* 90 (1961), pp. 33–49.

[38] D. Pereira. *WindRose.m*. 2015. URL: http://www.mathworks.com/matlabcentral/fileexchange/47248-wind-rose/content/WindRose.m.

[39] D. Reese and C. Melvin. "Smart LDAR: Pipe Dream or Potential Reality?" 7 (2006), pp. 1–20.

[40] D. R. Robinson, R. Luke-Boone, et al. "Refinery evaluation of optical imaging to locate fugitive emissions." *Journal of the Air & Waste Management Association (1995)* 57.7 (2007), pp. 803–810. ISSN: 10962247. DOI: 10.3155/1047-3289.57.7.803.

[41] S. Saunier. *Personal communication*. July 2, 2015.

[42] S. Saunier. *Quantifying Cost-effectiveness of Systematic Leak Detection and Repair Programs Using Infrared Cameras*. Tech. rep. Clean Air Task Force, 2014. URL: http://www.carbonlimits.no/PDF/Carbon_Limits_LDAR.pdf.

[43] J. Seinfeld and S. Pandis. *Atmospheric Chemistry and Physics*. 2nd. Hoboken, NJ: Wiley and Sons, Inc, 2006.

[44] M. Svrcek. *plotfixer.m*. 2001.

[45] *US Energy Information Administration Natural Gas Data*. 2015. URL: http://www.eia.gov/dnav/ng/hist/n9050us2a.htm (visited on 05/08/2015).

[46] *US Natural Gas Wellhead Price. US Energy Information Administration Natural Gas Data*. 2015. URL: http://www.eia.gov/dnav/ng/hist/n9190us3A.htm (visited on 10/06/2015).

[47] A. E. Wells. "Results of Recent Investigations of the Smelter Smoke Problem". *Journal of Industrial and Engineering Chemistry* 9.7 (1917), pp. 640–646.

A Appendix

A.1 Version history

Version 1.0 Released October 2015.

A.2 Leak production rate estimation

One of the estimates of the leak production rate is based on the VOC emissions reported by Carbon Limits for an average gas well battery in the first year following a repair program. We made several assumptions in order to convert the VOC emissions for one year into a leak production rate. Those assumptions are outlined here.

The first assumption is that the leakage grows linearly from zero throughout the year. In that case the leakage production rate is calculated according to Equation A.1, where $\frac{dL}{dt}$ is the rate of leakage production and μ_L is the average leakage rate over the time period ΔT.

$$\frac{dL}{dt} = 2\frac{\mu_L}{\Delta T} \tag{A.1}$$

Carbon Limits reports that $\mu_L = 1.8$ tVOC/y-well battery for $\Delta T = 1$ y, so the average leak production rate is 3.6 tvoc/y^2-well battery.

We estimate the total gas emitted by equation A.2. M_i is the molar mass of species i, and N_i is the number of moles of species i.

$$L_{\mathrm{CH_4}} = L_{VOC} \times \frac{M_{\mathrm{CH_4}}}{M_{\mathrm{VOC}}} \times \frac{N_{\mathrm{CH_4}}}{N_{\mathrm{VOC}}} \tag{A.2}$$

The average molar mass of VOCs is not reported in the dataset. We estimate M_{VOC} based on Equation A.3. $\mathbb{Z}_{\mathrm{VOC}}$ is the set of species in the VOCs and x_i is the mole fraction of species i in the VOCs. The VOC mole fractions x_i are assumed to match the speciation reported for natural gas leaks in API 4589 [5].

$$M_{\mathrm{VOC}} = \sum_{i \in \mathbb{Z}_{\mathrm{VOC}}} M_i x_i \tag{A.3}$$

The exact ratio of moles VOC to moles CH_4 is unknown ($N_{\mathrm{VOC}}/N_{\mathrm{CH_4}}$). However, Carbon Limits published VOC ratios used in their analysis for several types of gas. Based on personal communication with Carbon Limits [41],

we assume that all of the VOC emissions reported here come from sweet gas with a VOC mole fraction of 7.5%. As we have throughout this report, we approximate the total leakage as 100% CH_4.

Using the above assumptions, we calculate a leakage production rate of 1.28 g-CH_4/y^2-well battery or 1.1×10^{-3} g-CH_4/s-d-well battery. Finally, we need to estimate the number of wells per well battery. Carbon Limits reports that up to 15 wells can be included in a single battery, but data do not exist to define the average number of wells per well battery. We estimate the average number of wells per well battery based on data from the Fort Worth study. The study reports up to 13 wells in the well batteries visited, with an average of 2.9 wells per battery. Dividing our per-well battery production rate by 2.9 wells per battery we arrive at the leak production rate estimate of 3.8×10^{-4} g/s per well per day given in Subsection 3.4.2.

A.3 Flue gas versus CH_4 buoyant plume models

Flue gas and CH_4 plumes are both buoyant. Flue gas is buoyant because it enters the atmosphere at an elevated temperature, while CH_4 is buoyant because CH_4 has a lower molar mass than molecular nitrogen or oxygen. FEAST applies established flue gas buoyant plume models to CH_4. Here, we explore the difference between buoyancy due to the temperature of a gas and buoyancy due to the molar mass of a gas.

A plume that rises due to its temperature (a *thermal plume*) is distinct from a plume that rises due to its molar mass (a *chemical plume*) because its buoyancy declines due to thermal conduction and radiation in addition to entrainment of atmosphereic gases into the plume. We first show that convective mixing has the same effect on buoyancy in both thermal and chemical plumes, and then assess the significance of radiative and conductive cooling.

A.3.1 Convective mixing in buoyant plumes

As a buoyant plume entrains atmospheric air it becomes more dense. Equation A.4 gives the average specific volume of a plume that consists of a chemical with specific volume v_1 [m^3/kg] and entrained atmospheric gases with a specific volume v_0 [m^3/kg]. X is the volume fraction of the plume made up by the chemical plume [m^3 chemical/m^3 entrained gases].

$$v_{plume} = v_1 X + (1 - X)v_0 \tag{A.4}$$

Similarly, Equation A.5 gives the average specific volume of a thermal plume that has an initial temperature T_1 and entrains atmospheric gases at temperature T_0 before the gases equilibrate. We apply the ideal gas law to calculate the specific volumes, where R is the ideal gas constant [J/K-mol], P is the pressure (constant in both the plume and the atmosphere) [Pascal (Pa)], and M is the

molar mass of the molecules in both the plume and the atmosphere [kg/mol].

$$v_{plume} = \frac{T_1 R}{PM} X + (1 - X)\frac{T_0 R}{PM} \tag{A.5}$$

After the entrained gas equilibrates the plume is characterized by one temperature T given by Equation A.6, where c_p is the heat capacity of the gas [$c_p \approx 1 \times 10^3$ J/kg-K].

$$T = \frac{T_1 X M c_p + T_0(1 - X) M c_P}{X M c_p + (1 - X) M c_P} = T_1 X + T_0(1 - X) \tag{A.6}$$

Equation A.7 for the equilibrium plume specific volume easily follows from the ideal gas law:

$$v_{plume} = \frac{TR}{PM} = (T_1 X + T_0(1 - X))\frac{R}{PM} \tag{A.7}$$

$$\tag{A.8}$$

Comparison of Equations A.5 and A.7 shows that the specific volume of the plume does not change as the entrained gas equilibrates. Considering the ideal gas law we recognize that Equation A.5 can also be written in the form of Equation A.9

$$v_{plume} = v_1 X + (1 - X) v_0 \tag{A.9}$$

Comparison of the expressions for the specific volume of a thermal plume (Equation A.9) and a chemical plume (Equation A.4) shows that the effect of entrained gases is the same in both cases.

A.3.2 Conductive heat loss in thermal plumes

The conductive heat loss from the plume is estimated using the thermal conductivity of air, $k = 0.024$ W/K-m. We consider a differential volume dV given by Equation A.10, where r is a characteristic plume radius assumed to be 1 meter, and dx is a differential length along the axis of the plume [m].

$$dV = \pi r^2 dx \tag{A.10}$$

The rate of heat loss from dV is $k 2\pi r dx (T_1 - T_0)/l$, where l is the characteristic length that the heat must be transferred over. Dividing by the heat stored in the plume yields Equation A.11, where Q is the amount of heat that must be transferred out of the plume in order for it to reach atmospheric temperature [J] and ρ is the density of the plume ($\rho \approx 0.6$ kg/m^3 at a representative flue gas temperature of 600 K).

$$\frac{\delta Q}{Q} = \frac{k 2 \delta t}{l c_p \rho R} \tag{A.11}$$

In FEAST,we are interested in the behaviour of the plume within 10 meters of the leak source, or the first 10 seconds of plume dispersion. Choosing δt =10s and letting $l = r = 1m$, we find $\frac{\delta Q}{Q} = 0.1\%$. We conclude that heat loss due to conduction is negligible over the time scales of interest. Therefore, the fact that the density of a chemical plume does not change due to conduction of heat will not distinguish it from a thermal plume in the first 10 seconds after emission.

A.3.3 Radiative heat loss in thermal plumes

We estimate the heat loss from a thermal plume due to radiation based on the emissivity of flue gases. The emissivity ϵ is estimated following the method provided by [34]. We assume a flue gas composition of 10% H_2O, 10% CO_2, and 80% N_2. We find the total emissivity at a temperature of 600K and an approximate plume thickness of 1 m to be 0.25. As in the previous subsection, we consider a time interval of 10s and a flue gas temperature of 600K, and we also assume an ambient temperature T_0 of 300K. With these assumptions we compute $\delta Q_{rad}/Q$ in Equation A.12.

$$\frac{\delta Q}{Q} = \frac{\epsilon \sigma T_1^4 2\delta t}{(T_1 - T_0)c_p r} \tag{A.12}$$

Based on the above equation we find that approximately 10% of the heat that must be transferred from the plume for it to reach ambient temperature is radiated out of the plume in the first 10 seconds. This estimate is an upper bound on radiative heat loss, because it does not account for a decrease in temperature of the plume as it disperses. Thus, we conclude that a chemical plume may maintain up to 10% greater buoyant force after ten seconds of dispersion than would be expected of a thermal plume. The use of thermal plume rise models is justified as a reasonable approach.

A.4 IR camera based detection modeling methods

The expression for the signal in a single pixel of an IR camera that is imaging a natural gas plume is given by Equation 3.22. For convenience we express Equation 3.22 here as a combination of Equations A.14 and A.13.

$$\Gamma = \int_\Lambda \Phi(x(s), y(s), z(s))ds \tag{A.13}$$

Γ is the concentration pathlength integral [g-m/m^3]. The signal in a pixel is assumed to be proportional to Γ:

$$Signal = \alpha\Gamma \tag{A.14}$$

In the following two subsections we discuss the meaning of the proportionality constant α, and the method used to compute the integral in Equation A.14.

A.4.1 IR camera signal estimation

The signal in each pixel of an IR camera used for gas detection is due to the difference between the radiation incident on the pixel in the presence and absence of a natural gas plume. Equation A.14 assumes that the plume is between the camera and the dominant source of IR radiation. This assumption is justified by the low emissivity of air in the IR spectrum–we expect sunlight reflected off of background surfaces or the surfaces themselves to provide the dominant IR light source in all cases with little interference from IR photons emitted by the air.

Air has a very high transmissivity. As a result most of the photons emitted from the background source would reach the camera in the absence of a natural gas plume. Conversely, CH_4 absorbs IR radiation very strongly. As a photon propagates through a natural gas plume, it may be absorbed by the plume and re-emitted in a random direction. Some photons will be diverted away from the camera, while others that initially were on a path to miss the camera will be redirected into it. If the camera is pointed at the dominant background source of IR photons, the net effect will be a reduction in the number of IR photons incident on the camera's sensors.

As long as the plume is optically thin (meaning that the probability of an individual photon being absorbed is much less than one), the fraction of photons that are absorbed and re-emitted in the plume is proportional to both the concentration of molecules in the plume and the width of the plume. As a result, the difference between the radiation incident on a pixel imaging an IR source through a plume and through air is proportional to the integral of the concentration along the ray imaged by the pixel.

The actual signal in the IR camera depends on the sensitivity of the pixel and the software used to analyze the image. We do not attempt to model that complexity. Instead we rely on the minimum detectable concentration pathlength estimated by [3], and aggregate many properties of the camera in the empirical constant α.

A.4.2 Numerical concentration path length integral

The IR camera detection function in FEAST requires greater than 95% of the computation time in a FEAST simulation. Here we present the challenge in computing $\beta_{F,AIR}$ and $\beta_{F,MIR}$, and discuss the numerical methods implemented by FEAST.

In order to compute $\beta_{F,AIR}$ or $\beta_{F,MIR}$ we need to determine whether more than 10% of pixels have a Γ above the detection criteria in the best image that the camera will take during a survey. Given a frame rate of 60 frames per second, a survey time of 20 seconds, and 100×100 pixels per frame, each leak is associated with 1.2×10^6 Γ values. With 10^3 leaks at each time step and 3×10^1 surveys per simulation, the total number of concentration pathlengths associated with an AIR program simulation is on the order of 10^{10}. Needless

to say, it is important to compute Γ quickly, and to reduce the total number of computations whenever possible. We first discuss the method implemented to compute Γ, and then go on to discuss methods for reducing the number of computed images.

We define an expression for the concentration of CH_4 downwind of a leak ($\Phi(x, y, z)$) in Equation 3.9. Unfortunately, computing the integral of Φ along an arbitrary ray through the plume is difficult. In some cases an approximate analytic solution exists. For example, if we consider a path that passes through the center of the plume from directly above, at a location sufficiently downwind of the leak source that the plume buoyancy is negligible, and from a height sufficiently far above the center of the plume that the concentration of CH_4 is negligible behind the camera, Γ reduces to an integral of the standard form given by Equation A.15.

$$\int_{-\infty}^{\infty} \exp(-s^2/2\sigma)ds \tag{A.15}$$

Even if the camera is lowered to a location within the high concentration region of the plume, and the pixel of interest does not image the centerline of the plume, a change of coordinates can reduce the integral to the error function $\int_0^x \exp(-t^2)dt$ available in standard computing packages. However, we are interested in paths that are close the leak source and have a significant component parallel to the central axis of the plume. In this case the concentration pathlength integral does not fit the form of any standard function that we are aware of and we resort to numerical integration techniques.

We estimate Γ by a quadrature method indicated by Equation A.16. $\mathbb{Z}_Q$ is the set of points used in the quadrature method and Δs is the step size between quadrature points.

$$\Gamma \approx \sum_{i \in \mathbb{Z}_Q} \Phi(x_i, y_i, z_i)\Delta s \tag{A.16}$$

There are several properties of Φ that we use to speed up the quadrature calculation. First, we know the basic shape of the concentration profile, and that allows us to estimate the location along each integral path that has the highest concentration. Furthermore, we know that the AIR program will always image plumes looking straight down from above, and we assume that the same is true for the MIR program for simplicity[1]. FEAST identifies the approximate location of the highest concentration point on each integration path analytically, and limits $\mathbb{Z}_Q$ to a range centered on that point.

The range of $\mathbb{Z}_Q$ is chosen based on the width of the plume. More explicitly, the plume thickness in the vertical direction σ_z is calculated for the x position of the highest concentration point on the integration path. $\mathbb{Z}_Q$ then ranges

[1] In actuality, some plumes will be viewed from the side or from below in the MIR program. Allowing for these variations would increase the complexity of FEAST significantly, but have little effect on the end result because the plumes are nearly symmetric about their centerline.

over a distance of $8\sigma_z$, and may be cut short if the range goes upwind of the leak source, below ground level, or behind the camera. Due to the smooth concentration profile given by Φ we are able to estimate Γ with just 15 calculations of Φ. The signal from any pixels that image an area completely upwind of the leak source is immediately set to zero.

Even with 15 evaluations of Φ per Γ calculation, simulating every image taken in the AIR module would require 10^{11} calculations of Φ. We reduce that requirement by only simulating the best images of each plume that the program would produce. Given the flight path indicated in Figure 3.13, the full width of every leak will be included in an image at some point during the flight survey. Whether the plume is to one side of the image or directly below the camera makes very little difference in the Γ value associated with each pixel: the angle of the pixels changes slightly, but they still pass through the bulk of the plume. Therefore only considering images centered downwind of the leaks in the detection function has little impact on the results.

The exact location of the best plume image downwind of the leak depends on the leak size and the atmospheric conditions downwind of the leak. We do not have a fully deterministic method for determining where the best image will be taken. Instead, FEAST first considers an image with the leak source at the very upwind edge of the image. If the leak is not detected in that image, the camera is moved downwind by one-half of an image width and a new image is simulated. This process repeats until the plume is either detected, or the number of pixels above the detection limit begins to decline. Using this method two images are typically sufficient to determine whether the leak will be detected. The veracity of the method was tested by considering a random sample of 720 leaks and comparing the results of the high speed method with a method that searched for leaks from many more camera positions. The two methods classified the leaks as detectable or undetectable in exactly the same way (Figure A.1).

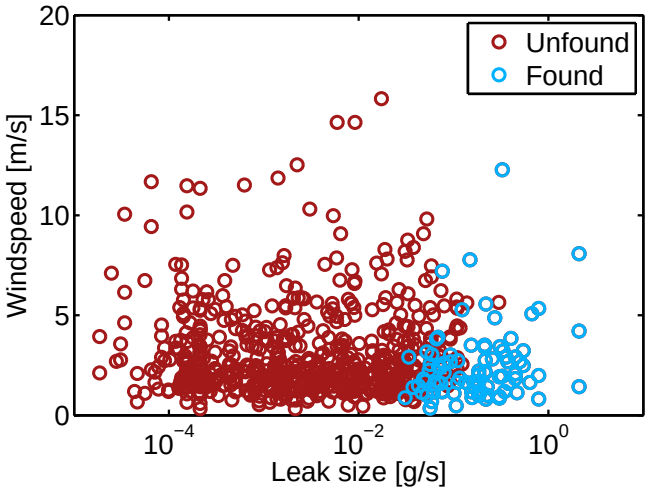

Figure A.1: Projection of random leaks into the flux-windspeed space. Leaks indicated in blue were found by both detection algorithms. Leaks indicated in red were not found by either algorithm. No leaks were found by one algorithm and not the other.