Integrated Diagnostics of Rocket Flight Control

Dimitry Gorinevsky*, Sikandar Samar†,
Honeywell Labs, Fremont, CA 94539
John Bain,
Honeywell Space Systems, Houston, TX 77058
and Gordon Aaseng,
Honeywell Space Systems, Glendale, AZ 85308

Abstract— This paper describes an integrated approach to parametric diagnostics demonstrated in a flight control simulation of a space launch vehicle. The proposed diagnostic approach is able to detect incipient faults despite the natural masking properties of feedback in the guidance and control loops. Estimation of time varying fault parameters uses parametric vehicle-level data and detailed dynamical models. The algorithms explicitly utilize the knowledge of fault monotonicity (damage can only increase, never improve with time) where available. The developed algorithms can be applied to health management of next generation space systems. We present a simulation case study of rocket ascent application to illustrate and validate the proposed approach.

Table of Contents

1. Introduction

This paper is focused on the developement of algorithms for vehicle health management (VHM) of a launch vehicle in closed-loop flight including vehicle dynamics, flight control actuators, sensors, navigation, and propulsion system. Flight control related faults can have severe impact on crew and vehicle safety during ascent. Safety margins could be improved by early detection of incipient faults (prognostics) to enable timely mission decision.

We are evolving a software-based capability for fault estimation by fusing cross-vehicle parametric data without introducing any additional sensors. The approach re-uses dynamical models available for the GN&C (Guidance, Navigation, and Control) system design and analysis. We demonstrate

the approach by using simulated telemetry data for a launch vehicle of Space Shuttle class. Faults seeded in the simulation are subsequently estimated by the VHM algorithms to validate their performance. The estimated fault parameters include air drag change from aerodynamic surface damage. This could model leading edge damage like that sustained in the Columbia Accident STS-107 mission. We also consider estimation and trending of such parameters as propulsion performance, thrust vectoring actuator/gimbal wear, and a drift in one of GN&C sensors (pitch angle). These faults are choosen as plausible representative faults that demonstrate the detection algorithm effectiveness. Development of a practical VHM system would require an additional careful analysis and engineering of the fault models in the VHM algorithms.

The fault estimation capabilities described in this paper will integrate smoothly with vehicle health management systems that use a test and diagnosis methodology. Systems such as the Honeywell's Boeing 777 Central Maintenance Computer (CMC) integrate discrete Buit-in-Test (BIT) data using vehicle-level diagnostic models to provide a substantial added value [14].

Adapting this approach to space systems, we continuously monitored simulated ISS telemetry data, generated test results from that data, and processed the test results through a model-based reasoner [1]. The system is expected to dramatically improve fault isolation and facilitate discrimination of the root cause from Caution & Warnings and sensor data, analysis that is performed manually by mission controllers.

Advanced testing methods are needed to detect systems dynamics and incipient faults. In this work we integrate parametric sensor data using vehicle-level models to estimate fault parameters for prognostics and incipient fault diagnostics. The challenge is to estimate incipient faults and trend system degradation hiding behind dynamical variation, feedback guidance and control, and noise of the sensor signals. We address this by using accurate dynamical vehicle models and optimal statistical estimation of fault condition. These results will significantly extend the capabilities of diagnostic reasoners by providing information about additional types of faults that can't be obtained through subsystem BIT and simple limit-checking.

We envisioned that initially the algorithms demonstrated in

---

this work could be deployed on-ground in a mission control center. The algorithms would use a stream of telemery data coming from a vehicle and provide on-line estimation of the fault parameters for mission management decision support. For the next generation space vehicles, the algorithms could be also deployed in on board avionics to support greater autonomy.

This work is a case study demonstrating fesibility of estimating fault parameters during an ascent of a space launch vehicle. At the same time, the technical approach presented herein is innovative as well.

There is a large body of work on model-based parametric diagnostics using detailed models of system dynamics. Much of this work was historically associated with controls community and considers linear dynamical systems. For examples of classical parametric estimation and identification methods applied to diagnostics see [6], [7], [11]. Modern systems theory methods applied to designing dynamical observer filters for fault estimation include H_2, H_∞, and such, e.g., see [3], [13].

The approach of this paper is related to the cited work in spirit but is different in a few respects. First, in the problem at hand the telemetry data is sampled at a high rate whereas the estimation update is relatively slower. This results in a multi-rate estimation problem. Second, the application problem considered in this work is inherently nonlinear. We use nonlinear models for computing prediction residuals and then linearize the problem with respect to the faults similar to how it is done in [5]. Third, and the most important difference is that we take some of the faults to be monotonic, i.e., they describe deterioration that can only get worse with time, never improve. The optimal statistical estimates for such faults can be computed in a batch mode. We make use of all the available data at any instant as opposed to a standard Kalman Filter recursive solution. The monotonic fault approach extends that of [9], [16].

There has been a recent focus on constrained state estimation as discussed in [8], [9], [15]. These techniques allow using a broad range of nonlinear filtering models. Some recent work has also discussed in depth the implementation of receding horizon filters implementing constraint problem solutions, which are most useful for an on-line implementation, e.g., see [4].

In this work, some of the estimated faults are modeled to be monotonic similar to [9]. Some other faults are modeled as non-monotonic, or simply constant. This results in a multi-rate estimation problem with a multi-variable mixture of fault parameters subject to constraints. We estimate the unknown faults by numerical optimization of the log-likelihood function. The formulation and optimization of the log-likelihood loss index reflects fault signature models, measurement noise statistics, and fault evolution models. Using optimization-

based estimation allows us to achieve flexible modeling, incorporate knowledge that damage conditions might only grow worse with time for some faults, and also make estimates robust to modeling errors. The optimization problem is a convex Quadratic Programming (QP) problem and a solution can be computed in efficient, scalable way using the state of the art solvers. Such optimization can be embedded into the on-line computations.

The paper is organized as follows. Section 2 introduces the application problem and gives an overview of the different faults and subsystems under consideration. In Section 3 we describe the residual based estimation approach. Section 4 provides a mathematical formulation for the estimation problem which serves as an input to the solver. The results of the estimation are given in Section 5.

2. APPLICATION PROBLEM

Rocket ascent simulation

The diagnostic algorithms developed in this paper are demonstrated through a simulation study of the ascent of the Shuttle-class vehicle depicted in Figure 1 [2]. We simulate the vehicle dynamics, kinematics, guidance, navigation & control (GN&C), propulsion, and consider some representative system faults. The detailed simulation model and the models
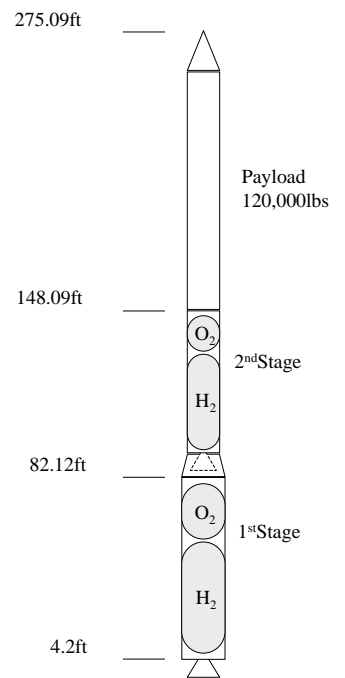

Figure 1. Block Stations

used for fault detection are developed in-house and described in detail elsewhere [10]. The measured states and control history of the simulation are logged as simulated telemetry data. This data is subsequently used for validating the fault estimation algorithms developed in Section 3.

As a part of the case study, a simulation model of the launch vehicle ascent was developed. This is a simplified model of a Space Shuttle class vehicle that is based on data published in the open literature [2]. The detailed engineering simulation used by NASA for the Space Shuttle has in excess of 100 states and thousands of parameters. Details of such complex simulations are understood by teams of people. To demonstrate our diagnostics algorithms, we choose to create a more manageable yet representative example. The rocket launch simulation set up in this study has on the order of 10 states, a few dozen parameters, and is easily understood by a single person. Though the details of the simulation are not presented here, the highlights are presented below.

The dynamical model of this simulation consists of a set of ordinary differential equations (ODE) that are numerically integrated. This ODE system is stiff, since the timescales for rocket motion and fast GN&C actuators are very different. Also, the magnitude of the variables runs many orders of magnitude, i.e. the control torque is required to move an engine bell producing 5×10^6 lb thrust for angle tracking of $\approx 10^{-3}$ deg.; while the final value of the achieved altitude is $\approx 5 \times 10^5$ ft.

We simulate a two-stage vehicle with liquid fuel (H_2 and O_2) delivering a medium payload to low Earth orbit. We consider in-plane dynamics only for planar equatorial flight that terminates in a 0° inclination orbit. The modeling includes variation of mass, center of gravity, and moment of inertia with the propellant expenditure. The aerodynamic model is borrowed from an asymmetric vehicle (the first stage drag minimum is at small positive angle of attack) [2]. The model assumes a non-rotating spherical Earth and uses an inverse square gravitational field and an exponential atmosphere. In the diagnostics example to follow, the first stage of the trajectory is primarily considered since the nonlinearities of the aerodynamic model are most predominant while still in the thick atmosphere. Expansion to the full launch sequence would be straightforward.

As depicted in Figure 2, a GN&C system is implemented as a part of this simulation. For this trajectory-following guidance
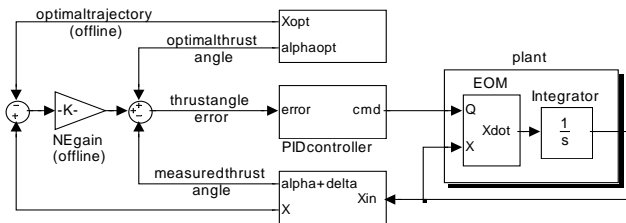

Figure 2. Block Diagram

and control scheme, the engine bell is gimbaled to attain the desired pitch angle while the thrust is maintained at 100% (it is a function of altitude and stage). We assume full-state measurements for the navigation model. Neighboring Extremal

(NE) closed-loop guidance uses a point mass simulation to calculate the minimum fuel trajectory [2]. NE guidance is implemented as full-state proportional feedback, which does not steer back to the original optimal trajectory. Instead, at each guidance calculation, if the vehicle has deviated from the optimal path, NE guidance provides the desired control for the optimal trajectory from that point to the desired endpoint. In the reported results, a PID controller tuned using trial-and-error is used for the main engine gimbal actuator to keep the vehicle on the NE trajectory.

As shown in Figure 3, the state variables for this vehicle simulation are:

- κ is the downrange angle measured in rad
- h is the vehicle's altitude in ft
- v is the vehicle's velocity in ft/s
- γ is the flight path angle measured in rad
- θ is the vehicle's pitch angle measured in rad
- δ is the engine gimbal angle measured in rad
- ω_e is the engine rotational rate in rad/s
- ω is the vehicle's rotational rate in rad/s

where h is positive up, κ is increasing as the rocket flies downrange, and ω_e is positive in the direction pictured as positive δ. Using this state definition, the equations of motion for
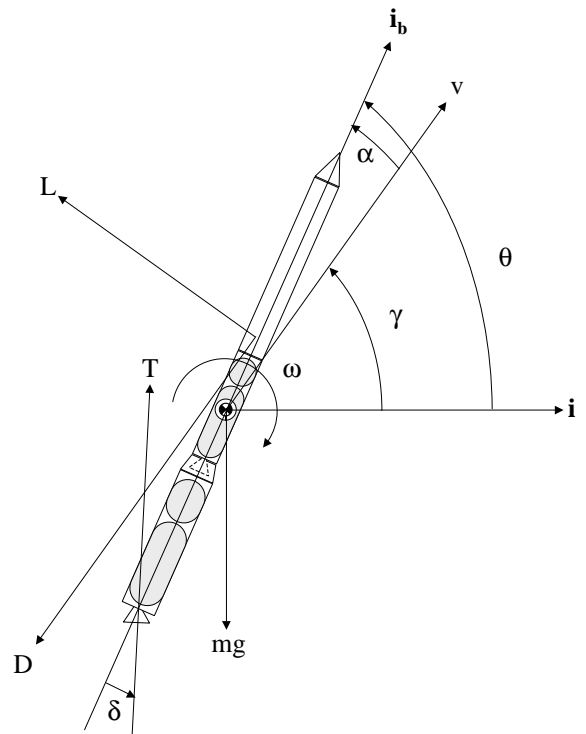

Figure 3. Rocket

this vehicle are [10]

$$\dot{\kappa} = \frac{v \cos \gamma}{r_s + h}, \qquad (1)$$

$$\dot{h} = v \sin \gamma, \qquad (2)$$

$$\dot{v} = \frac{1}{m + m_e}\left[(T - m_e l_e \omega_e^2)\cos(\alpha + \delta)\right.$$
$$- (m + m_e)g\sin\gamma - m_e\omega^2 l\cos\alpha$$
$$\left. - m_e l_e \dot{\omega}_e \sin(\alpha + \delta) + ml\dot{\omega}\sin\alpha - D\right], \qquad (3)$$
$$\dot{\gamma} = \frac{1}{(m + m_e)v}\left[(T - m_e l_e \omega_e^2)\sin(\alpha + \delta)\right.$$
$$- (m + m_e)g\cos\gamma + L + m_e l_e \dot{\omega}_e \cos(\alpha + \delta)$$
$$\left. - ml\dot{\omega}\cos\alpha - m_e l\omega^2\sin\alpha\right] + \frac{v\cos\gamma}{r_s + h} \qquad (4)$$
$$\dot{\theta} = -\omega, \qquad (5)$$
$$\dot{\delta} = \omega - \omega_e, \qquad (6)$$

with the rotational accelerations

$$J^*\dot{\omega} + J_c\dot{\omega}_e\cos\delta = J_c\omega_e^2\sin\delta$$
$$+ \quad Q - \frac{mlT\sin\delta}{m + m_e}$$
$$+ \quad \left(l_{cp} - \frac{ml}{m + m_e}\right)(L\cos\alpha + D\sin\alpha), \qquad (7)$$
$$J_e^*\dot{\omega}_e + J_c\dot{\omega}\cos\delta = -J_c\omega^2\sin\delta - Q$$
$$+ \quad \frac{m_e l_e}{m + m_e}\left[L\cos(\alpha + \delta) + D\sin(\alpha + \delta)\right]. \qquad (8)$$

In Equations (1) to (8)

$$J^* = J + \frac{mm_e l^2}{m + m_e}, \qquad (9)$$
$$J_e^* = J_e + \frac{mm_e l_e^2}{m + m_e}, \qquad (10)$$
$$J_c = \frac{mm_e ll_e}{m + m_e}, \qquad (11)$$
$$\alpha = \theta - \gamma. \qquad (12)$$

In Equations (7) and (8), Q is the torque to move the engine bell. The gimbal actuator is modeled as a first order actuator of the form

$$\dot{Q} = 20k(x_d - x_s) \qquad (13)$$

where x_d is the desired servo actuator position, x_s is the servo position, and k is the servo position-to-torque gain. Additionally, m is the rocket mass, J is the rocket inertia, l is the distance from the gimbal pivot to the rocket center of gravity, and l_{cp} is the distance from the gimbal pivot to the center of pressure. The engine bell has corresponding quantities m_e, J_e, and l_e. The lift and drag forces are L and D, thrust is T, and gravity is denoted g.

The simulation is PC based and programmed in Mathworks' Simulink (double precision) as a continuous time model with Runge-Kutta integration of the ODEs. The simulation covers the ≈ 371 sec. trajectory of ascent into an 80×150 nm equatorial orbit. The vehicle launch mass is 1.03×10^5 slugs, mass flow rate is constant in each stage, and staging occurs upon expending of first stage propellant at the time 153.54 s. The sensor and control data is logged at a 0.1 s sampling interval and saved into a data file. The simulation model includes an ability to add (seed) the faults as described below. The simulated telemetry data is subsequently used for validating the fault estimation algorithms that do not have an access to the seeded faults and should estimate them from the trajectory data.

Modeled faults

The algorithms developed in this work offer a great deal of flexibility in the estimation of parametric faults. These algorithms are capable of estimating constant, step, monotonic, and non-monotonic time-varying faults. The nature of the faults is not a limiting factor (though fault observability through the available data is). The particular faults chosen in this study were the ones providing a good demonstration of the approach capabilities and are not meant to represent a practical design of a vehicle health management system. They are *representative* of real faults that span several systems of concern within the closed-loop flight system.

We consider the following four parametric faults aggregated into a fault vector to be estimated:

$$f := \begin{bmatrix} \text{Thrust Loss, percent} \\ \text{Drag Increase, percent} \\ \text{Gimbal Sluggishness, percent} \\ \text{Pitch Sensor Offset, percent} \end{bmatrix} \qquad (14)$$

The first fault vector component, percentage of thrust loss, is related to the propulsion subsystem. It describes the degradation that occurs in the propulsion engine over a period of time. We model this fault as a monotonic one assuming that the efficiency of the rocket engine can only decrease (deteriorate) with time.

The second component of vector f describes an increase in the aerodynamic drag of the vehicle. The drag mostly impacts the dynamics during first stage ascent while the vehicle is still in the thick atmosphere. The importance of estimating this fault parameter can be related to the Columbia accident. The increase in drag might indicate damage to the aerodynamic (and heat shielding surfaces).

The third element in the fault vector is the percentage of sluggishness of the gimbal actuator. The gimbal actuator is used for vectoring the main engine thrust and is the primary control actuator. Gimbal sluggishness during a vehicle's launch may be caused by a pressure loss in the actuator's hydraulic system, some hydraulic valve problem, or in wear of of mechanical gears or bearings. This sluggishness may lead to a deviation from the desired trajectory since it effects the control system for the vectored thrust steering torque.

The fourth fault we consider in this work relates to drift of the pitch sensor in the GN&C system. The origin of this fault may have to do with some error in calibration of the sensor or in wiring induced noise or offset. This fault may lead to an incorrect value of the vehicle's pitch angle used by the guidance and control laws.

4

The developed simulation includes the faults of interest as parts of the propulsion subsystem, flight dynamics (GN&C) subsystem and the gimbal actuation subsystem models.

Diagnostics data availability

The 'telemetry' data stream used by the algorithms is produced by the simulation discussed in Subsection 2. This data is available at a 100ms sampling interval. At each sample, the data generated by the simulation consists of (i) vehicle state data (8 values), (ii) vehicle acceleration data (4 values), and (iii) gimbal servo data (2 values). At sampling time t these data make the 14-component data vector $x(t)$.

We assume vehicle rigid-body acceleration data is available as a part of telemetry. The accelerations are normally available to the vehicle GN&C system from on-board accelerometer or a navigation Kalman Filter estimator that could also use position and velocity data. The bandwidth of the acceleration measurement and estimation would be much higher than the 10 Hz sampling rate of the telemetry data. The vector of accelerations has the following components:

- $\dot{v}$ is the vehicle's vertical acceleration in ft/s^2
- $\dot{\gamma}$ is the flight path angle rate measured in rad/s
- $\dot{\omega}_e$ is the engine's rotational acceleration in rad/s^2
- $\dot{\omega}$ is the vehicle's rotational acceleration in rad/s^2

The gimbal actuator servo data include two values:

- x_{act} is the gimbal actuator position measured in rad
- x_d is the gimbal actuator servo command measured in rad

Figure 4 illustrates telemetry data obtained for the first stage ascent of a launch vehicle in the simulation example that is further discussed in Section 5.

3. Estimation Problem Statement

In this work we use *multirate* data processing assuming that the fault estimation algorithms take longer than the 100 ms sampling interval to execute. As the new data vectors $x(t)$ from (simulated) telemetry become available, they are stored till the next cycle of algorithm execution. The described timeline of the data processing logic is illustrated in Figure 5. The results of Section 5 have been obtained with the estimation update algorithms running every 15 seconds to process the data including the new 150 vectors $x(t)$ accumulated through that time.

The algorithms' outputs are the fault estimates in the form of time functions. We validate algorithms by demonstrating that these estimates match the time-varying faults seeded in the simulation when generating the 'telemetry' data.

The fault estimation algorithms perform on-line computations using the residuals and fault signatures from the prediction models as inputs and providing estimates of unknown fault parameters as a function of time at the outputs. Whereas

the fault estimates must be computed on-line, the fault signatures and other data used by the estimation algorithms can be pre-computed off-line for the planned ascent trajectory as is explained below. The on-line estimation is cast as a convex optimization problem that can be solved very efficiently.
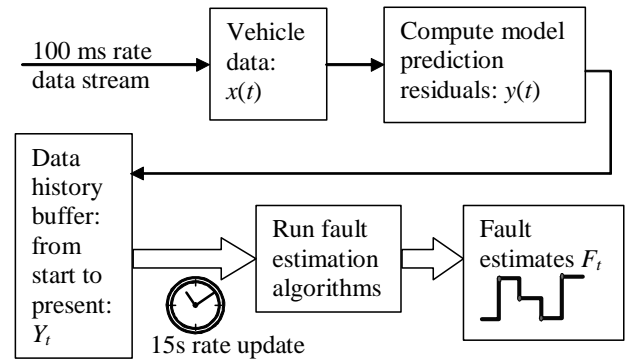

Figure 5. Fault estimation timeline

Prediction Model

The prediction model is a key part of the estimation algorithms. It computes parity relationships between the measured variables. The prediction model outputs should be zero in the absence of faults and reflect the fault parameters. The prediction models for the three subsystems: propulsion, GN&C, and gimbal actuation, are developed separately and then integrated. Though the prediction model is logically different from the simulation model discussed in Section 2, both are based on the same dynamical knowledge. One major difference between the simulation and prediction models is that the simulation includes closed-loop GN&C while the prediction uses the applied control input to compute the expected (predicted) system outputs (such as vehicle rigid body accelerations) and is thus independent of the control algorithms.

For each of the three subsystems, prediction models take the relevant telemetry data and computed data from other subsystems as inputs and predict some outputs (other telemetry data). The difference between the predicted and the actually observed outputs yields the prediction residuals. The prediction models used in this work also accept the four faults as additional inputs. This is needed for modeling fault signatures.

The GN&C prediction model takes as its inputs the data vector $X(t)$ including the vehicle state and the gimbal actuator position. It also inputs the thrust value predicted by the propulsion subsystem model. The outputs are three residuals for predicting (i) vertical acceleration, (ii) flight angle rate, and (iii) pitch acceleration.

Consider now the prediction model for the gimbal actuation subsystem. The hydraulic actuator of the main engine gimbal suspension is used for vectoring the main engine thrust and provides the control for the GN&C system. The gimbal ac-
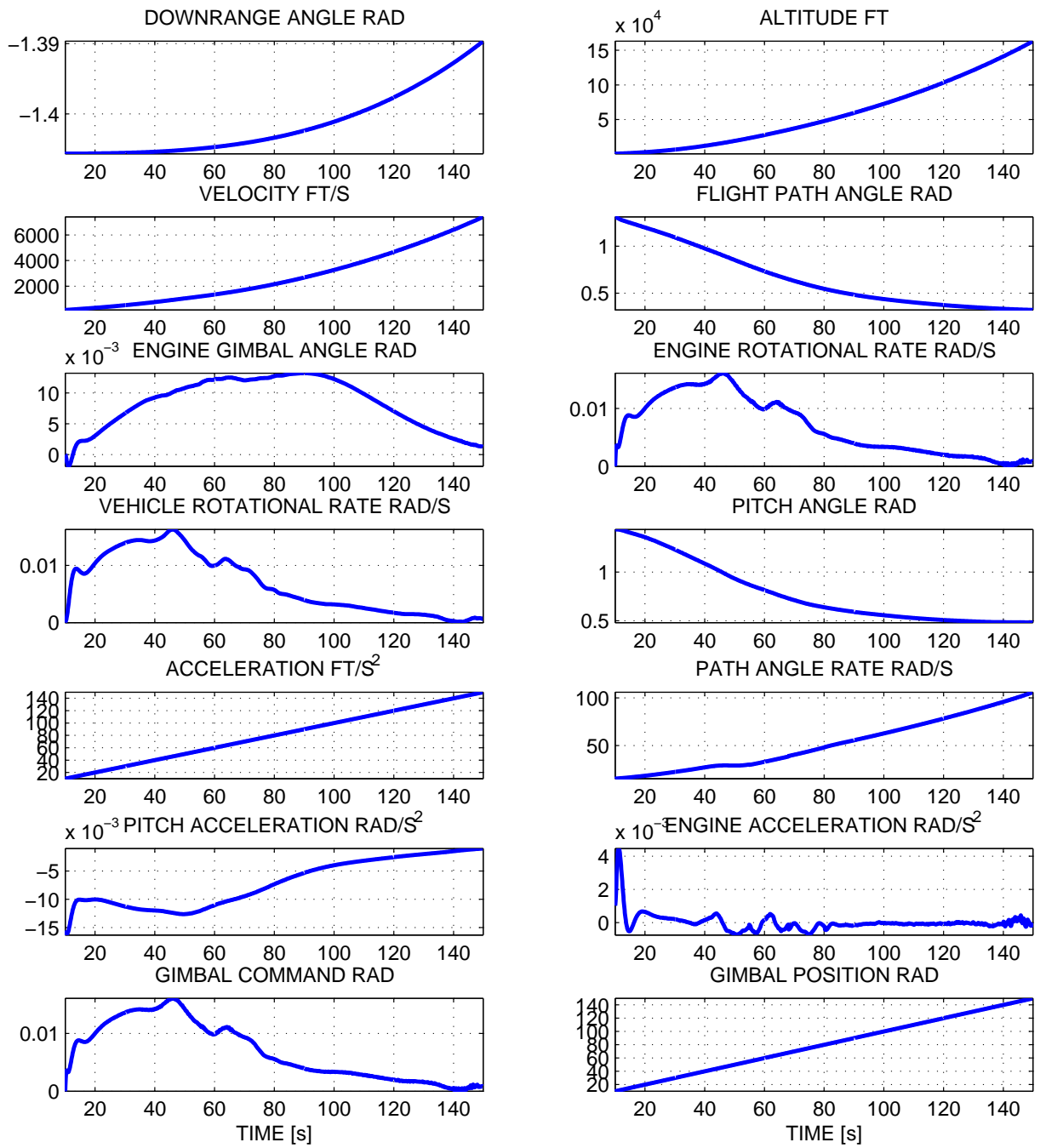
Figure 4. Telemetry Data for First Stage Ascent Simulation

tuation includes a servo-system for tracking the desired vectoring angle. The dynamics are linearized by the closed-loop feedback and are assumed to have the form

$$\dot{x}_{act} = (x_d - x_{act})\omega_f, \tag{15}$$

where ω_f is the bandwidth of the closed-loop feedback control servo system which guides the vehicle along the desired trajectory. The reduction of the bandwidth parameter ω_f indicates the gimbal actuator sluggishness.

The predictive residual is computed such that it is zero if x_d and x_{act} satisfy the model (15). The nonzero residual should reflect the change in ω_f. The difficulty is that differentiation of the signal $x_{act}(t)$ is required to verify if (15) holds. To avoid increasing the noise in a finite difference estimation of the derivative, we use a smoothing differentiator. The model prediction residual is computed as follows

$$r_{\text{gimbal}}(t) = \frac{s}{s + \tau_s} x_{act} - \frac{1}{s + \tau_s} x_d, \tag{16}$$

where $r_{\text{gimbal}}(t)$ is the servo rate residual (in rad/s), s is the Laplace variable (differentiator operator), τ_s is a smoothing filter pole (time constant) and the two transfer functions describe the filtering operators applied to the two respective signals. The two filters in the r.h.s. of (16) have proper and strictly proper transfer functions respectively and can be easy implemented. Note that if (15) is satisfied, then the residual in (16) is exactly zero. Applying a smoothing filter is a linear transformation of the signal. It does not offset the residual, just brings about a filtering delay while enabling differentiator implementation.

The overall residual vector $y(t)$ is a combination of the GN&C subsystem residuals and the gimbal subsystem residual

$$y(t) := \begin{bmatrix} \text{Vertical acceleration residual, ft/s}^2 \\ \text{Flight angle rate residual, rad/s} \\ \text{Pitch acceleration residual, rad/s}^2 \\ \text{Servo rate residual, rad/s} \end{bmatrix}. \tag{17}$$

The residual vector $y(t)$ is calculated at the same rate as the sampling rate for the telemetry data $x(t)$, which in our case corresponds to a sampling time of 100ms.

Residual-based Estimation

A non-zero residual vector in (17) indicates an off-nominal behavior and implies presence of faults. The integrated diagnostics algorithms use these residuals along with the fault signatures as inputs to determine the fault estimates. The fault signatures, also referred to as the fault sensitivities, provide a mapping between the unknown faults and the residuals. In many practical cases, including the one in hand, this mapping can be assumed linear. This allows us to efficiently solve the estimation problem by reducing it to a convex optimization problem.
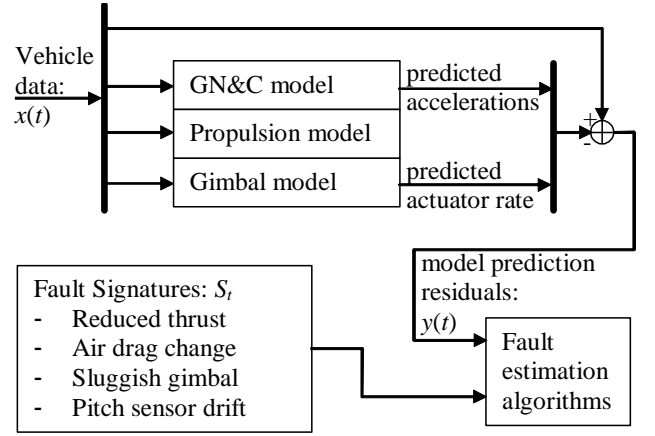

Figure 6. Model-based residual processing

When processing the data $x(t)$, the algorithms first use detailed prediction models for computing residuals - mismatches between model prediction and actually observed system outputs. The faults are then estimated from the residuals using the fault signatures (fault models). This is illustrated in Figure 6 and explained in more detail in the remainder of this section.

The residual data $y(t)$ is sampled at a high rate of 100ms. There are approximately 1530 samples in the residual vector for the 153 seconds of the first stage ascent duration. The estimation algorithm runs at a much slower rate, once every M high-rate samples. In our example the update is every 15 seconds, i.e., for every $M = 150$ samples of residual data. The choice of the update interval depends on a variety of factors including the efficiency of the algorithm and the hardware limitations of the on-board processors.

For the multi-rate system explained in the preceding paragraph, we let τ be the estimation update cycle. Then the residual data vector accumulated from the lift-off till the estimation update cycle number τ will be denoted as

$$Y_\tau = \begin{bmatrix} y(1) \\ \vdots \\ y(\tau \cdot M) \end{bmatrix} \tag{18}$$

The sampled-data estimation logic assumes that fault parameters $f(t)$ (14) are constant through each estimate cycle, i.e., $f(t) = F(n)$, for $M(n-1) < t \le Mn$. This assumption reduces the number of the fault values that are estimated and improves statistical averaging properties of the estimation scheme. At the same time, there is little loss of estimation performance in addition to the already accepted sampling time delay of the estimation update. The fault parameter vector accumulated from the lift-off till the estimation update cycle number τ will be denoted as

$$F_\tau = \begin{bmatrix} F(1) \\ \vdots \\ F(\tau) \end{bmatrix} \tag{19}$$

The algorithm objective is finding the unknown fault parameter vector F_τ. This requires relating F_τ to the available residual data vector Y_τ. If the faults don't change the underlying system dynamics substantially, we can linearize dependence between the residuals and the unknown faults. This assumption allows us to express the relationship in the form

$$Y_\tau(|F_\tau) \approx S_\tau \cdot F_\tau + \zeta_\tau \qquad (20)$$

where S_τ is the fault sensitivity matrix, ζ_τ is the noise vector which accounts for all the modeling uncertainties such as propulsion generated vibration, thrust fluctuations, turbulence, unmodeled flexible dynamics of the rocket, fuel slosh, model parameter errors, sensor measurement noise, etc. Note that if $\zeta_\tau = 0$ in (20) (no noise, no modeling error), then the residual in the absense of fault $Y_\tau(|0) = 0$, as it should be.

The dependence of the residual data vector Y_τ on the faults F_τ in general is not available in an analytical form for computing the sensitivity (Jacobian) matrix S_τ. This dependence is however encoded into our prediction model and we can compute the sensitivity matrix by a finite difference method. This is done by simply incrementing each component of the fault vector F_τ and then running the prediction model with the corresponding fault inputs to compute pointwise values of the residual data vector $Y_\tau(|F_\tau)$. The finite difference estimate of the columns of S_τ is obtained by normalizing increments of the observed residual vector change. The sensitivity matrix computation may be performed off-line prior to the launch of the vehicle. The pre-computed sensitivity matrix may then be stored and later used by the estimation algorithms during on-line computations. Such a computation of S_τ assumes that the vehicle will closely follow the nominal trajectory during the flight. If the actual state of the vehicle doesn't match the desired trajectory, then there will be an inaccuracy in the computed sensitivity matrix.

An alternate method is to compute the sensitivity matrix online using the actual vehicle state obtained from the sensors. In this approach the nominal prediction model is run alongside the prediction models with each of the fault inputs in an online setting using the actual vehicle state at that time of the flight.

The discussion above was about the matrix S_τ corresponding to the estimation update cycle τ. This matrix does not need to be computed from scratch at each update cycle. Instead it can be computed once for the terminal update cycle $\tau = T$ of the ascent. For any $\tau < T$ the matrix S_τ is a truncation of the matrix S_T (a $M\tau \times \tau$ submatrix of the $MT \times T$ matrix S_T).

Figure 7 shows the columns of the linear operator $S(\tau)$ for the update cycle $\tau = 4$, i.e., $45 \le t \le 60$). The operator is causal in the sense that since $y(t|F_\tau)$ does not depend on $F(\rho)$, $\rho > t/M$. It is also clear from the figure that the map is sparse in time because of the negligible influence of fault after-effect on the residual vector for the next cycle.
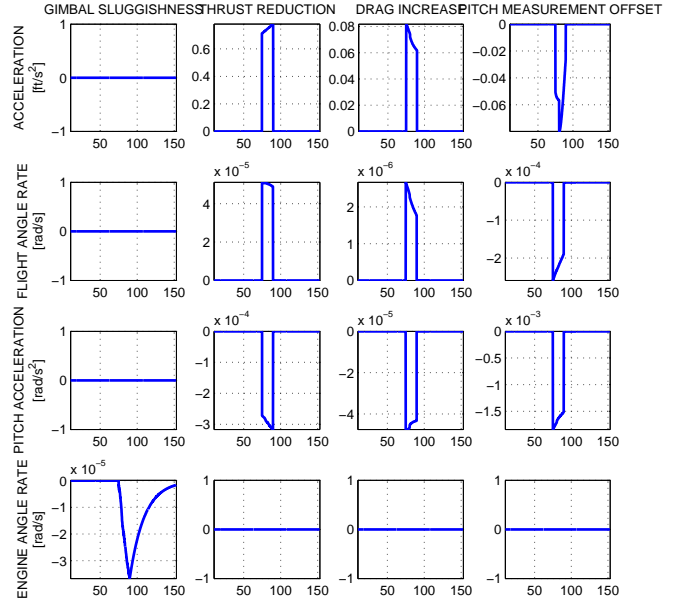

Figure 7. Fault signatures computed for an interval during the first stage ascend

4. SOLUTION APPROACH

Given the statistical models in (20), (24), the estimation problem is to find the unknown fault parameter sequence F from the residual data Y. Index τ is dropped, it can be anything or final

The Maximum Likelihood (ML) estimate of the unknown fault sequence is obtained by numerical optimization of the log-likelihood

$$J = -\log \mathcal{P}(F|Y) \to \min \qquad (21)$$

By using Bayes rule and independence of the fault vector components $F_k(\tau)$ in (24) for different k we obtain

$$\mathcal{P}(F|Y) = \text{const} \cdot \mathcal{P}(Y|F) \cdot \prod_{k=1}^{4} \mathcal{P}(F_k(\cdot)), \qquad (22)$$

where $F_k(\cdot)$ denotes the entire time series $F_k(1), \ldots, F_k(L)$.

We assume that in the model (20), the noise $\zeta(\tau)$ is normally distributed zero-mean with covariance Q. This is a usual assumption leading to a least-square fit estimation. In that case we have $\log \mathcal{P}(Y|F) = -\frac{1}{2}(Y - SF)^T Q^{-1}(Y - SF)$.

Using the Bayesian conditional probabilities (22), we can thus present the loss function (21) in the form

$$J = \frac{1}{2}(Y - SF)^T Q^{-1}(Y - SF) + \sum_{k=1}^{4} J_k \quad (23)$$

The terms $J_k = -\log \mathcal{P}(F_k(\cdot))$ in the loss index (23) depend on the nature of each fault in the fault vector F.

We use a random walk model for the probabilistic modeling of the unknown fault parameter sequence, i.e.,

$$F_k(n + 1) = F_k(n) + \xi_k(n), \quad \xi_k(n) \sim p_k(x), \quad (24)$$

where $F_k(n)$ denotes the respective fault component at the update cycle n and $\xi_k(n)$ is the process noise driving the evolution of F_k. We assume that $\xi_k(n)$ are mutually independent for different k and n and have probablity distributions with the density functions $p_k(x)$. Then

$$J_k = \sum_{\tau=2}^{L} -\log p_k \left(F_k(n) - F_k(n-1) \right). \quad (25)$$

The initial fault state vector $F(1)$ is assumed to be unknown (its components $F_k(1)$ do not contribute to J_k in (25)). The distribution $p_k(x)$ of the random variable $\xi_k(n)$ reflects a prior knowledge about the fault and a fault evolution model. In this work we consider three types of such models: the faults that are non-monotonic, monotonic, or constant.

Case I: Non-monotonic Faults— Assume the driving noise $\xi_k(n)$ of the fault evolution model (24) to be zero mean normally distributed with covariance r_k. This yields a standard random walk model and $-\log p_k(x) = x^2/(2r_k)$. In this case J_k adds a quadratic penalty term to the loss index (23) and the problem becomes an unconstrained generalized least squares estimation. A recursive solution of such problem can be computed using a Kalman Filter formulation. In the example to follow, pitch sensor offset is assumed to be a non-monotonic fault.

Case 2: Monotonic Faults— In some instances the faults are known to increase (or decrease) with time, e.g., because the accumulation of an irreversive damage. For such monotonic faults we consider $\xi_k(n)$ to have an exponential distribution with width λ_k such that $-\log p_k(x) = x/\lambda_k$ in (25) with a constraint that $x \geq 0$. The performance index (23) now includes a linear penalty in J_k and should be minimized subject to the constraints $F_k(n + 1) \geq F_k(n)$. This yields a constrained quadratic programming (QP) problem which can be solved efficiently using convex optimization solvers. More detail on monotonic fault modeling and estimation can be found in [9]. In the example to follow, thrust loss and drag increase are assumed to be monotonically increasing faults.

Case 3: Constant Faults— A fault may describe an unknown condition that does not change after the liftoff (or staging). Such fault may be assumed to be constant. In this case the loss function (23) can be assumed to have $J_k = 0$ for respective k while being subject and to an equality constraints of the form $F_k(n + 1) = F_k(n)$. In the simulation example to follow, gimbal sluggishness is assumed to be a constant fault.

The above described models of fault evolution are of course empirical models and a careful engineering judgement should be exercised when deciding which one to use. The probability distribution parameters in these models can be considered as

tuning parameters, similar to how the noise covariances are selected in the Kalman Filtering practice. If the real faults do not exactly follow the assumed models, the estimation algorithms should still produce meaningful results compatible with the made assumptions (e.g., monotonicity).

Estimation Algorithm

As discussed in the previous section, the problem of minimizing the performance index (23) with or without the constraints is a convex optimization problem. Several efficient routines are available to solve such QP problems. To enable embedded implementation, we use a solver based on an interior point method. This high performance convex solver provides an estimate of the fault vector F as a solution to the constrained (or unconstrained) QP problem given the fault sensitivity matrix S and the residual data vector Y. The solver is implemented in Matlab, the simulation and prediction models were developed in Simulink. The solver uses sparse arithmetic and exploits the problem structure of the problem under consideration to efficiently compute the fault estimates. It can estimate fault parameters that fall in any of the three categories discussed above by minimizing the appropriate loss index.

The covariances of the noises $\zeta(t)$ and $\xi(n)$ are used as tuning parameters in the optimization solution. They are empirically chosen to obtain good fault estimates. The efficiency of the computation largely depends on the individual problem structure, i.e., the number of constraints in the problem and the sparsity structure of the arrays involved. The estimates will in general be computed more efficiently if the fault signatures are calculated off-line.

For the launch vehicle ascent example, the estimation algorithms consist of two parts. The first part are off-line pre-launch preparation computations. During this phase each of the considered faults is in turn seeded in the simulation model. The simulation is then run to obtain the residuals corresponding to this fault. These residuals correspond to fault signatures and allow to compute the fault sensitivity matrix S using the procedure described given in Section 3.

After the off-line preparation is complete, the optimization-based estimation is run on-line as the telemetry data is received. These on-line algorithms make the second part of the estimation.

5. ESTIMATION RESULTS

The specific fault estimation problem considered in this paper has a four element fault vector given in (14). For the purpose of a simulation we decided to assign different values to these four faults as shown in Figure 9. The sluggishness of the gimbal actuator was assumed constant during the first phase of the flight. A value of 20 percent was seeded for this fault. The percentage of thrust loss in the propulsion subsystem was assumed to grow with the expenditure of fuel. A gradual increase in thrust loss from 1 to 3.5 percent was seeded. For the
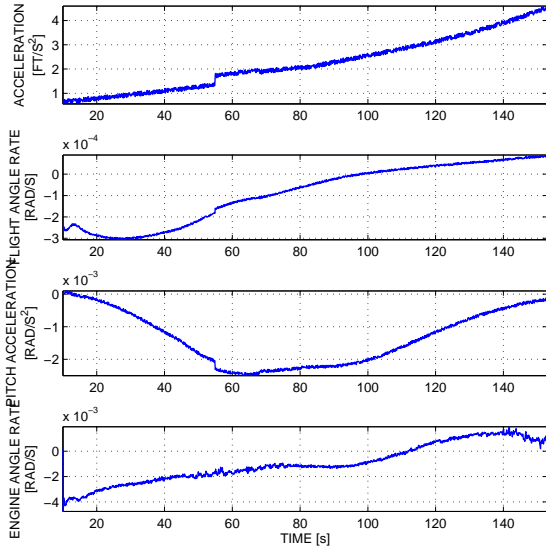
Figure 8. Prediction Residuals for First Stage Ascent

Figure 9. Comparison of Seeded Faults and Fault Estimates

drag a four percent step increase after 60 seconds of the flight was seeded in the simulation model. This may correspond to a sudden increase due to the aerodynamics surface damage, such as a piece of the fuel tank insulation foam falling off and damaging the leading edge of the wing in the Columbia accident. The pitch sensor drift in the GN&C sensor is assumed to vary sinusoidally between 0.1 to 0.8 percent during the course of the first stage of the flight. To verify the validity of the estimation approach, the seeded faults were selected to span all three cases discussed in the previous section: non-monotonic, monotonic, and constant fault. The simulation produced 'telemetry' data that was logged and provided to the fault estimation algorithms. First, the prediction model is run to compute the residuals for this data. The resulting residual data vector is shown in Figure 8. These nonzero residuals indicate the presence of faults. To make the simulation more realistic, a uniformly distributed uncorrelated random noise was added to the telemetry signals. The noise magnitude makes about 5-10% of the residuals. The residuals were processed by the multirate optimization-based estimation algorithm described in Section 3. As mentioned earlier the data in this example is sampled every 100 ms. The estimation algorithm runs every 15 seconds producing a total of 15 estimates during the 153 second of the first stage ascent. To enable this fault estimation, we pre-compute the sensitivity matrix S as explained in Section 3.

Before running the estimation calculations, the inputs are scaled. This is necessary, since the variables values measured in different physical units might differ by many orders of magnitude from one another. As an example, the vehicle climbs to an altitude of approximately $5 \cdot 10^5$ ft whereas the angle tracking is on the order of about 10^{-3} radians. To avoid poor conditioning of the sensitivity matrix, the estimates of all variables are scaled and converted into nondimensional units.
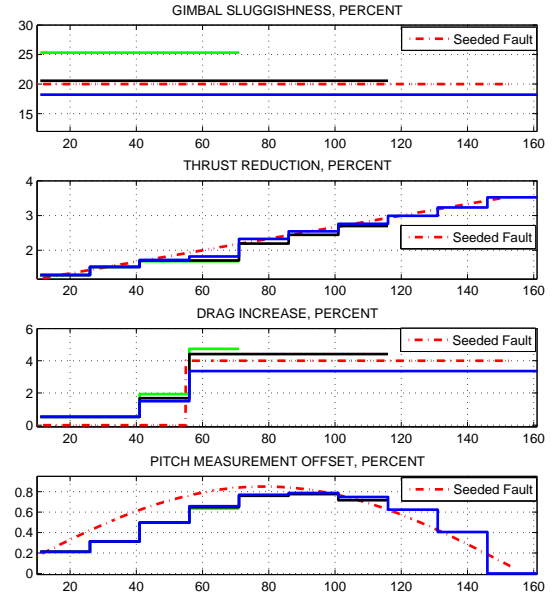
The scaling was selected empirically to make all the nondimensional variables about the same order of magnitude and improve problem conditioning.

For validation, the obtained estimates are compared against the faults that were actually seeded in the initial simulation. Figure 9 shows the estimates computed at $t = 70$, 115, and 150. As seen in the plots the estimates improve with time, as more data are accumulated, and match the unknown faults reasonably well despite the noise and the unaccounted non-linearity. The estimates may be tuned further by running an update every 4 or 8 seconds instead of the chosen 15 seconds interval.

6. DISCUSSION OF INTEGRATED DIAGNOSTICS

The multivariate trending and detection of parametric faults is a part of a holistic approach to Integrated Vehicle Health Monitoring (IVHM). This paper has described a methodology for detecting a class of faults that will contribute substantially to an overall understanding of the vehicle health state. By combining the results of multivariate detection with all of the subsystem Fault Identification Detection and Recovery (FDIR) logic, subsystem BIT, limit checking, simple trend detection, and other symptom detection methods, a diagnostic system can be made substantially more capable.

An envisioned advanced integrated diagnostic system of a space vehicle will present to the users as complete a picture of the health state of the vehicle as practicable [1]. The health state is a description of the ability of the systems, subsystems and components to perform their designed function. Each vehicle element is described as nominal or off-nominal, along

with the degree of degradation of off-nominal elements. This paper demonstrates an ability to detect fault conditions before they have fully developed. This allows to characterize incipient failure modes; rather than waiting for a major degradation in functionality, the onset of the deterioration is identified. As with other more familiar failure modes, the identification of the root cause likely requires correlation between several indications. For example the "gimbal sluggishness" indication may correspond to other symptoms in the hydraulic, electrical or other systems indicated by BIT messages. These correlations may be completed using a vehicle level reasoner.

In order to understand the root cause of a failure, it is necessary to correlate all of the effects of the failure and trace the effects back to the root cause. In a system using only BIT, it is likely that not all of the information needed to isolate the cause will be available. Adding sensors could possibly remove ambiguity in the diagnostic information; however, each sensor adds weight, power draw and complexity. The ability to get more information out of existing sensors using advanced analytical techniques as we have described is a significant advantage toward the goal of total health state knowledge.

7. CONCLUSIONS

This paper described an approach to mutivariable integrated diagnostics and trending of parametric faults. The approach was discussed and illustrated for a case study of flight control system diagnostics during space lauch vehicle ascent. The approach might be useful in many other aerospace applications. The developed algorithms require using detailed models for computing fault residuals and signatures. Such 'controls' type models are typically available as a part of control system design and analysis for aerospace systems. The estimation is based on embedded optimization and offers flexibility in types of faults that can be estimated: constant, time-varying, and monotonic. Using state-of-the-art convex solvers allows achieving high computational performance. This implies both ground telemetry-based and on-line embedded implementation of the algorithms can be feasible.

8. ACKNOWLEDGEMENTS

REFERENCES

[1] G. Aaseng, "Blueprint for an Integrated Vehicle Health Management System," *IEEE 20th Digital Avionics Systems Conference Proceedings*, Daytona Beach, Florida, Oct 14–Oct 18, 2001

[2] J. Bain and J. Speyer, "Robust neighboring extremal guidance for the advanced launch system." *Proceedings of the AIAA Guidance, Navigation and Control Conference*, Monterey, CA. August, 1993.

[3] R.K. Douglas, and J.L. Speyer, "H-infinity bounded fault detection filter," *AIAA Journal of Guidance, Control, and Dynamics*, Vol. 22, No. 1, 1999, pp. 129–138.

[4] G. Ferrari-Trecate, D. Mignone, and M. Morari, "Moving horizon estimation for hybrid systems," *IEEE Transactions on Automatic Control*, Vol. 47, No. 10, 2002, pp. 1663–1676.

[5] S. Ganguli, S. Deo, and D. Gorinevsky, "Parametric fault modeling and diagnostics of a turbofan engine," *IEEE Conference on Control Applications*, Taipei, Taiwan, September 2004.

[6] J. Gertler, *Fault detection and diagnosis in engineering systems*, New York: Marcel Dekker, 1998.

[7] J. Gertler, "Survey of model-based failure detection and isolation in complex plants," *IEEE Control Systems Magazine*, Vol. 8, No. 6, 1988, pp. 3–11.

[8] G.C. Goodwin, M. Seron, and J.A. De Dona, *Constrained control and estimation: An optimization approach*, New York: Springer-Verlag, 2004.

[9] D. Gorinevsky, "Monotonic regression filters for trending gradual deterioration faults," *American Control Conference*, Boston, MA, June 30–July 2, 2004.

[10] D. Gorinevsky, J.R.Bain, and G.A. Aaseng "Parametric diagnostics of flight control and propulsion for rocket ascent," *Technical Report*, Honeywell Aerospace Electronics Systems, Fremont, CA, January 2004

[11] R. Isermann, "Supervision, fault-detection and fault-diagnosis methods–an introduction," *Control Engineering Practice*, Vol. 5, No. 5, 1997, pp. 638–652.

[12] E.O. Nwadiogbu, M.J. Roemer, and G. Bloor, "Development of diagnostic and prognostic technologies for aerospace health management applications," *Proceedings of IEEE Aerospace Conference*, Vol. 6, pp. 3139–3147, Big Sky, Montana, March 2001.

[13] R.J. Patton, J. Chen, and B.S. Nielsen, "Model-based methods for fault diagnosis: some guidelines," *Transactions of The Institute of Measurement and Control*, Vol. 17, No. 2, 1995, pp. 73–83.

[14] G. Ramohalli, "The Honeywell on-board diagnostic and maintenance system for the Boeing 777," *Proceedings of 11th IEEE/AIAA Digital Avionics Systems Conference*, pages 485–490, October 1992.

[15] C.V. Rao, J.B. Rawlings, and D.Q Mayne, "Constrained state estimation for nonlinear discrete-time systems: Stability and moving horizon approximations," *IEEE Transactions on Automatic Control*, Vol. 48, No. 2, 2003, pp. 246–258.

[16] S. Samar, D. Gorinevsky, and S. Boyd, "Moving horizon filter for monotonic trends," *IEEE Conference on Decision and Control*, Paradise Island, Bahamas, December 14–17, 2004.

Dimitry Gorinevsky is a Senior Staff Scientist with Honeywell Labs and a Consulting Professor of Electrical Engineering at Stanford University. He obtained a Ph.D. from Moscow State (Lomonosov) University. He worked on a broad range of decision and control system applications in US, Canada, Germany, and Russia. He published a book and 130+ technical papers. Dr. Gorinevsky is an Associate Editor of IEEE Transactions on Control Systems Technology. He is a recipient of Control Systems Technology Award and Transactions on Control Systems Technology Outstanding Paper Award, both of the IEEE Control Systems Society. For last several years, he works on health management systems.

Sikandar Samar is a Ph.D. candidate in the department of Aeronautics and Astronautics at Stanford University, CA. He works as a Research Assistant at the Information Systems Laboratory in the Department of Electrical Engineering at Stanford. He worked as a summer intern with Honeywell Laboratories. His research interests include large scale estimation via convex optimization and applications of systems and control theory. He obtained an M.S. degree in Mechanical Engineering from University of Illinois at Urbana-Champaign in 2003 with a focus in control systems.

John Bain is an engineer with Honeywell Space Systems in Houston, Texas. His research interests include guidance and control for a range of aerospace vehicles. He is the author of over 10 articles in the areas indicated above. Dr. Bain obtained an B.S. degree in electrical engineering from the University of Texas at Austin in 1985, and a Ph.D. in aerospace Engineering from the University of California, Los Angeles in 1997 where his research advisor was Jason Speyer. He is a member of IEEE and AIAA.

Gordon Aaseng is a Senior Staff Engineer with 14 years experience in aerospace engineering programs. He is currently Technical Director of Honeywell's VHM IR&D projects in space systems area and IVHM Integrated Product Team Lead, responsible for projects developing architectures and prototype systems for diagnostics, prognostics, and related health management technologies. He led Honeywell's development of the IVHM demonstration project currently installed in the Quest Lab at NASA JSC, and has given numerous IVHM demonstrations and presentations to NASA and industry. His experience with the ISS program includes leading the systems engineering for the MATE program used for development and qualification testing of ISS flight software and developing requirements and architecture for Mission Control Center systems at JSC. Prior to entering the engineering field, he was a U. S. Naval Officer and Aviator accumulating over 2500 flight hours. He holds a Master's Degree in Computer Science from Texas A&M University at Corpus Christi.