Document Title: Interpretation of Ignitable Liquid Residues in Fire Debris Analysis: Effects of Competitive Adsorption, Development of an Expert System and Assessment of the False Positive/Incorrect Assignment Rate

Author(s): J. Graham Rankin, Ph.D., Nicholas Petraco

Document No.: 248242

Date Received: September 2014

Award Number: 2010-DN-BX-K272

Report Title

Interpretation of Ignitable Liquid Residues in Fire Debris Analysis: Effects of Competitive Adsorption, Development of an Expert System and Assessment of the False Positive/Incorrect Assignment Rate.

Award Number: 2010-DN-BX-K272

Author /Principal Investigator	J. Graham Rankin, PhD, Professor (retired) Forensic Science Program Marshall University 1401 Forensic Science Dr. Huntington, WV 25701 Rankinj@marshall.edu 304-634-1532
Co-Principal Investigator:	Nicholas Petraco, Associate Professor, John Jay College of Criminal Justice City University of New York 524 West 59th street, New York, NY 10019 npetraco@gmail.com 917-584-3895
Administrative POC:	William Byrd Marshall University Research Corporation 401 11th Street Huntington, WV 25701 byrdw@marshall.edu 304-696-6324
Financial POC:	Jan Weece Forensic Science Center Marshall University 1401 Forensic Science Dr. Huntington, WV 25701 Weece@marshall.edu 304-691-8957

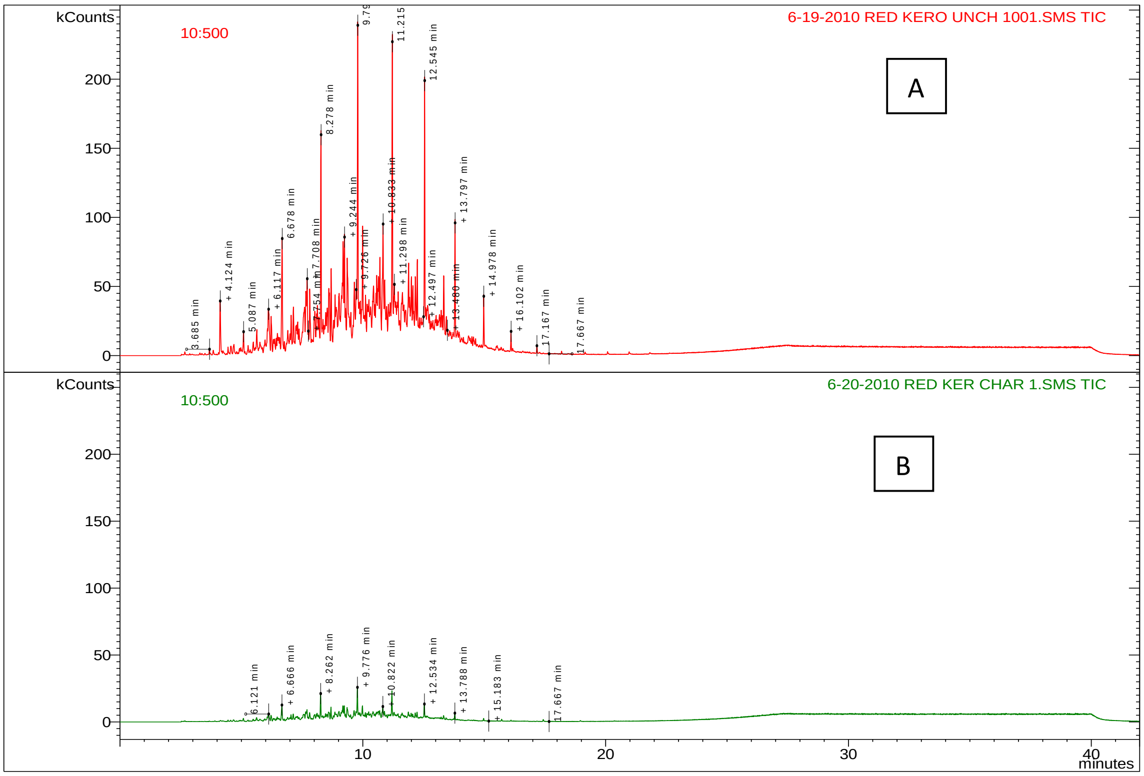


Figure 8. Red oak uncharred (A) and charred (B) comparison with kerosene spike.

Western red cedar charred substrates, when spiked with gasoline, showed a decrease in relative peak height in the entire chromatogram. However, both the charred and uncharred substrates had large gasoline overall response. The kerosene spiked samples, like many of the other substrates tested, had a decrease in relative peak height of about 50% when compared to the uncharred spike, noted in Figure 9.

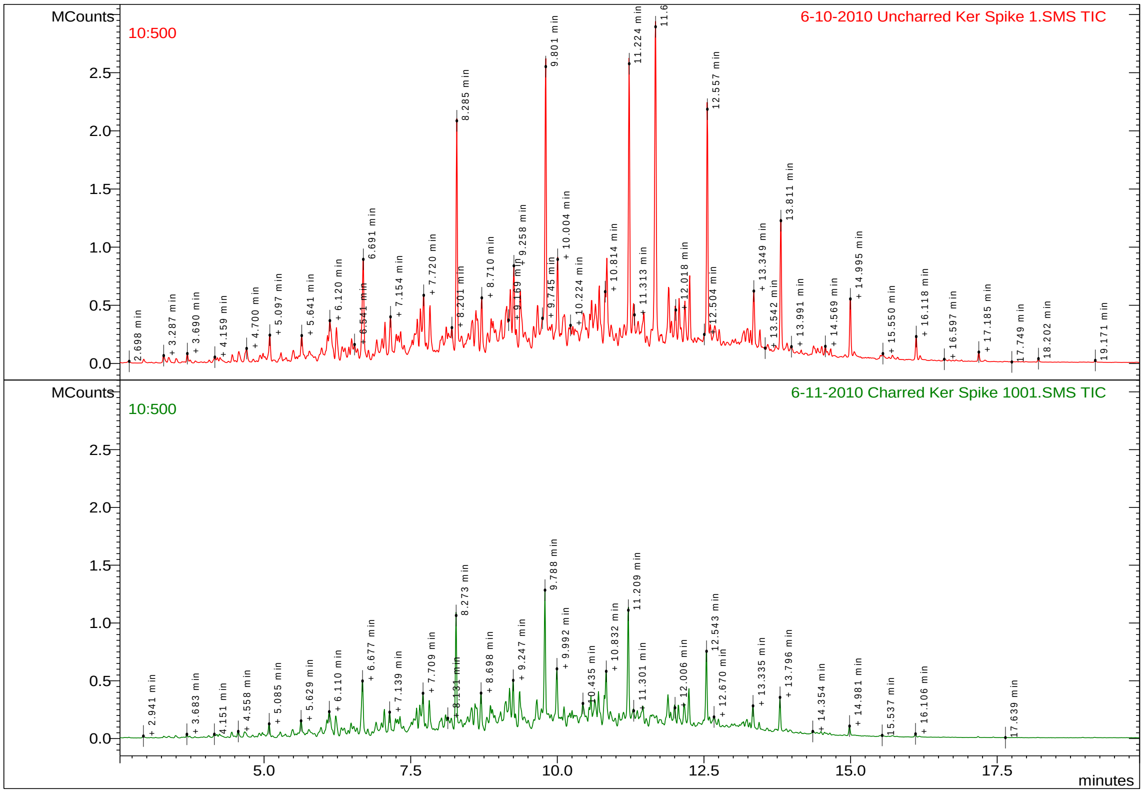


Figure 9. Western red cedar uncharred (A) and charred (B) comparison with kerosene spike.

Finally, the white pine substrate, which demonstrated interference peaks from the blank uncharred substrate, had those same peaks show up in the beginning of the chromatogram of the charred and uncharred substrate spike. Overall chromatographic response in the gasoline spike was smaller in the charred sample. In the kerosene spiked sample, the overall response was also decreased, and can be seen in Figure 10.

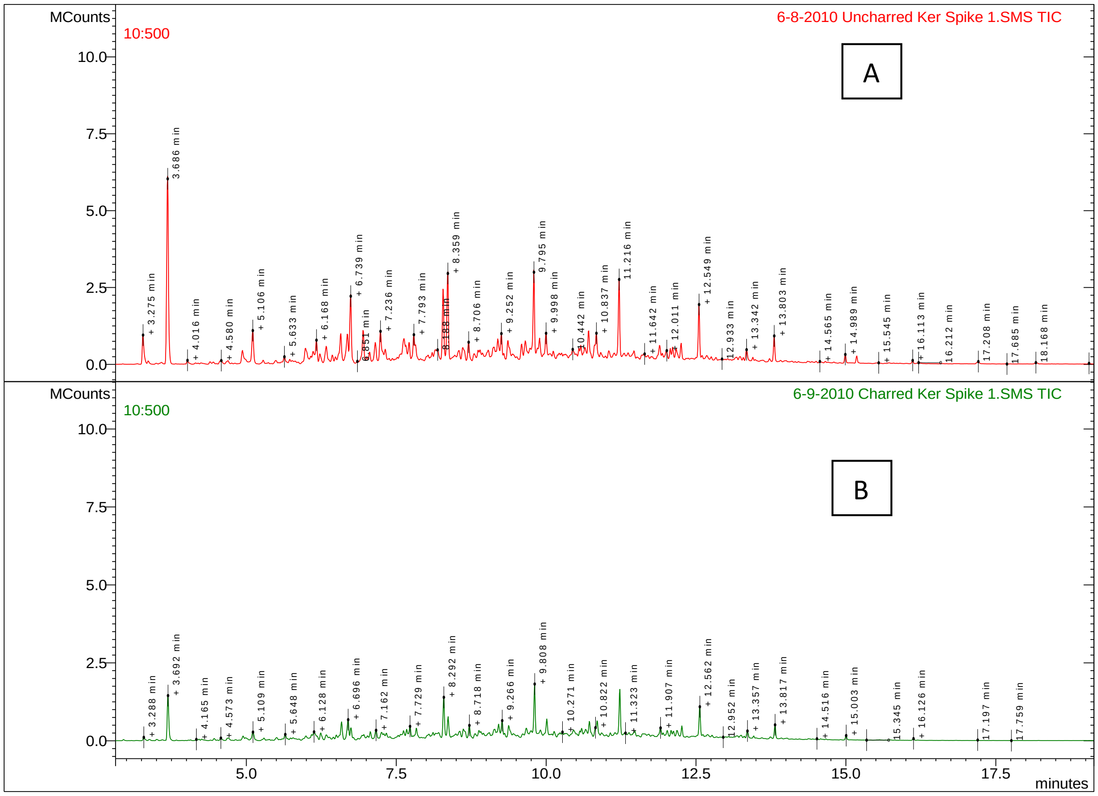


Figure 10. White pine uncharred (A) and charred (B) comparison with kerosene spike.

Variation with Percentage Charred

Two sets of wood, white pine and aspen, were used in this portion of the experiment to determine if varying the amount of charring would show a more significant decrease overall in the chromatogram response. The white pine and the aspen were both spiked with gasoline.

Comparing the chromatograms in the white pine substrate samples showed a decrease in overall response the more the wood was charred, as shown in Figure 11. In white pine substrates that were 20% charred compared to 40% charred, the overall chromatographic response decreased by 25%. However, when comparing the white pine substrates charred 40% to 60% charred, the chromatographic response was almost lost. The peaks that are present are about 20% the size of the peaks represented in the 40% charred chromatogram.

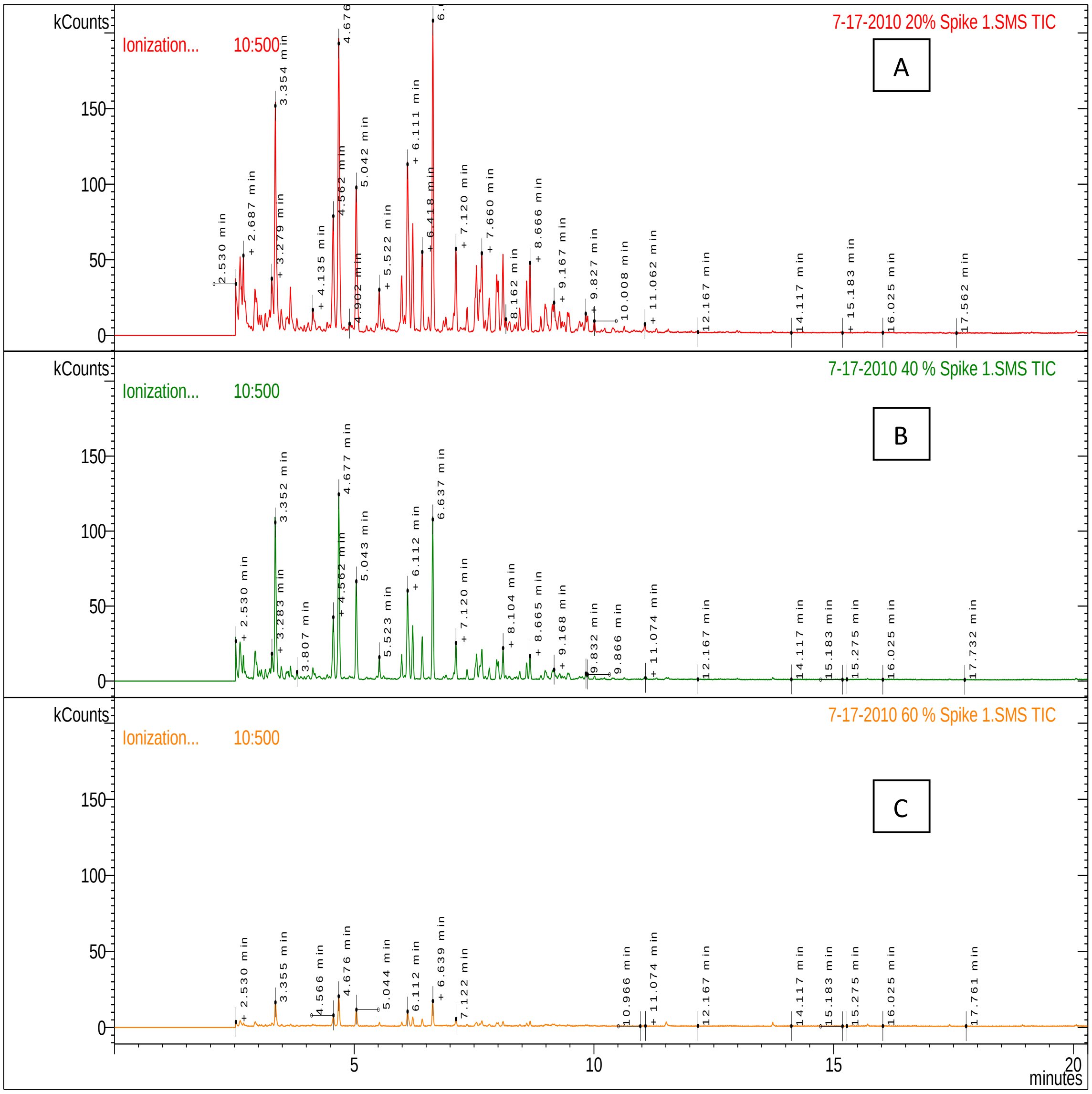


Figure 11. White pine percent charred comparison of 20% (A) 40% (B) and 60% (C) chromatogram spiked with gasoline.

In aspen, the more the substrate was charred, the smaller the overall response was observed, as well as some peaks becoming absent. Comparing the two substrates, the white pine showed more of a significant peak height loss between the varying levels. However, from the 20% charred to 40% charred, about 33% of the response was lost. Comparing the 40% charred to 60% charred about 50% of the response is almost lost, seen in Figure 12. The substrate charred and uncharred blanks showed no peaks present on the chromatograms that differed from the initial study substrate blanks of aspen.

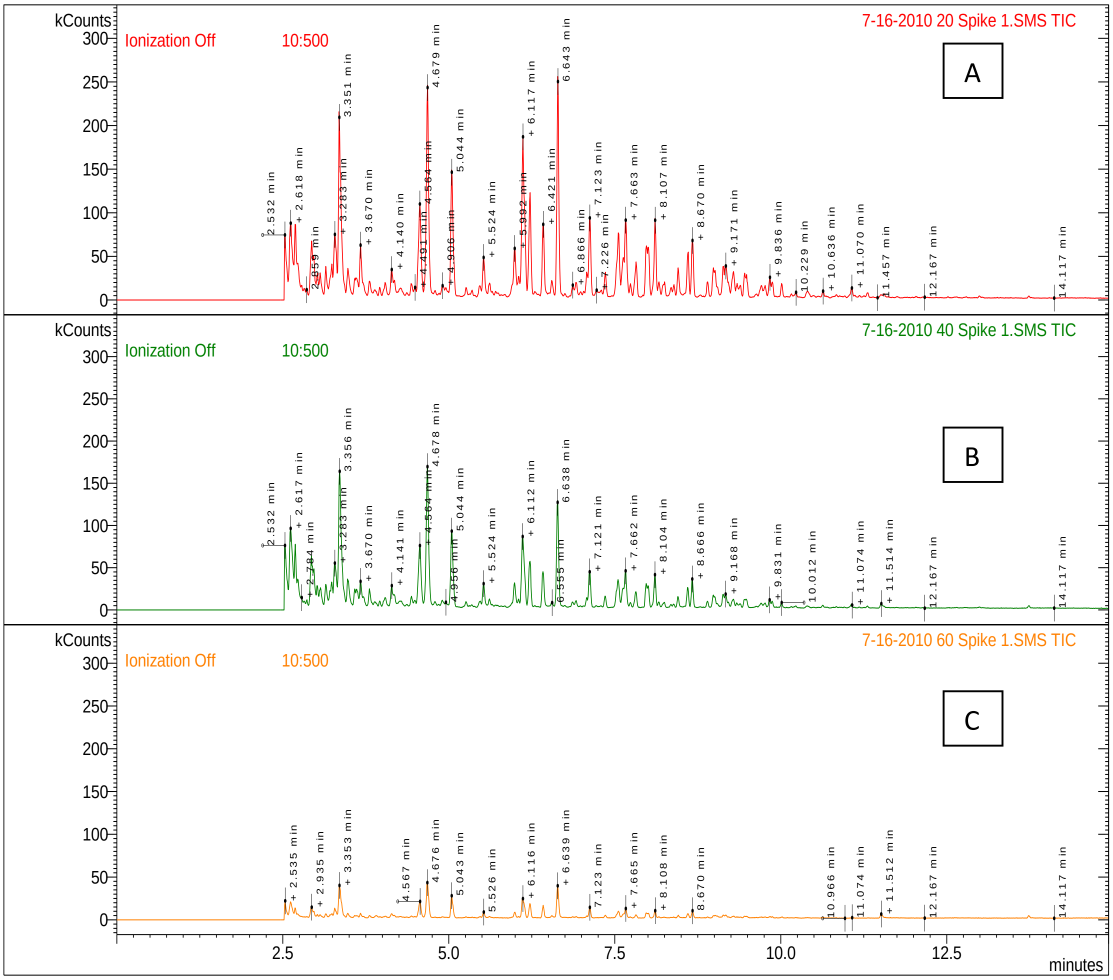


Figure 12. Aspen percent charred comparison of 20% (A) 40% (B) and 60% (C) chromatogram spiked with gasoline**.**

Results from differences in the substrate blanks between the charred and uncharred blanks could easily be explained by the wood having interferrents located on its surface. This conclusion is drawn from the absence of any peaks on the charred substrate blank. The peaks present could have come from a number of sources, for instance the manufacturing plant, during transportation, in a garage, basically in any area where the wood was kept for any duration of time. Since these peaks are absent in the charred substrate it could be concluded that the wood may not have any significant chemicals in it, or that once charred this chemical or compound is removed.

Based on the results obtained during this experiment, something is causing a decrease in overall response of the chromatograms. From chromatograms of the kerosene spiked samples (See Figure 10 vs Figure 11), a decrease in response of samples were present more significantly than a decrease in response in the gasoline spiked substrates. One of the main reasons this might be caused is due to the chemical make-up of the particular ignitable liquid. While kerosene is mostly composed of normal hydrocarbons, gasoline contains mostly aromatics with lesser amounts of iso- and cyclo-hydrocarbons with negligible normal hydrocarbons. Due to the nature of these compounds, it might be more difficult for the wood or the char to retain particular compounds in the kerosene samples, resulting in gasoline compounds to not have as a significant decrease in response.

Although competitive adsorption has been well known for some time [1, 2], the purpose of this study was to further eludicate the selective retention of some classes of compounds (i.e. normal hydrocarbons observed by Kelly [2]) as a function of substrate and amount of char. During the charring process it was noted that the more charred the substrate became, (due to the more cracks and nooks present in the wood), the greater the apparent surface area of char. With the increased surface area of char, the wood would have more area available to retain the compounds of the IL, and not releasing them into the headspace when heated. However, one could also argue that because there was more charred surface area, that the compounds competing for space on the activated charcoal had also increased. This means that the pyrolysis products released and the ignitable liquid were competing for space on the strip. Further studies potentially using another method of fire debris analysis might give more insight into this dilemma. Also, future studies, including using more wood samples in the varying the amount of percentage charred would be beneficial to determine if the wood itself is a variable.

Figure 13 demonstrates that there is a decrease in 1, 2, 4-Trimethyl Benzene found in gasoline, one of the main peaks used to determine and classify gasoline. Although this is not a conclusive trend that happened consistently, this does demonstrate the overall response after losing more weight in the charring process. Again, this might be due to the increased surface area present, causing less to be vaporized and collected on the ACS.

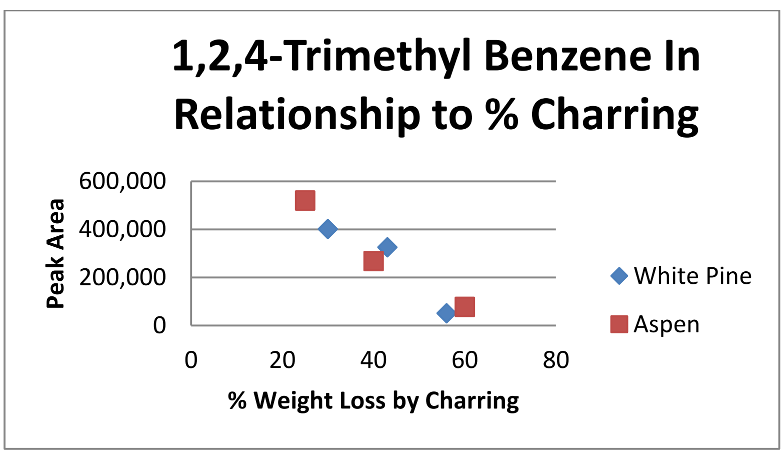


Figure 13. Relationship of percent charring relative to 1, 2, 4-TMB peak area.

After the initial study, additional substrates (yellow pine, polypropylene carpet and foam carpet pad) and an additional ignitable liquid (MPD = Charcoal lighter) were tested. Extracted ion chromatograms (EICs) were prepared for five different compound types to allow for E1618 classification. The EICs extracted were those of alkanes, cycloalkanes, alkylbenzenes, naphthalenes, and indanes. The ions (m/z) extracted from the TIC are presented in Table 5. The experimental chromatograms were then compared to previously run neat sample chromatograms of the same ignitable liquids to determine what differences, if any, were observed in the burned samples.

Table 5: Ions Extracted for Each Compound Class [11]

Compound Type	Ions (m/z)
Alkanes	57, 71, 85, 99
Cycloalkanes	41, 55, 69, 83
Alkylbenzenes	91, 92, 105, 106, 119, 120
Naphthlenes	128, 142, 156, 170
Indanes	117, 118, 131, 132

For yellow pine, several notable terpenes were present in the un-charred blanks (1S-α-pinene, 5.6 min; β-pinene, 6.2 min; 1,5-dimethyl-1,5-cyclooctadiene, 7.0min) as shown in Figure 14. In the charred samples, the terpene chromatographic peaks were greatly diminished (Figure 15).

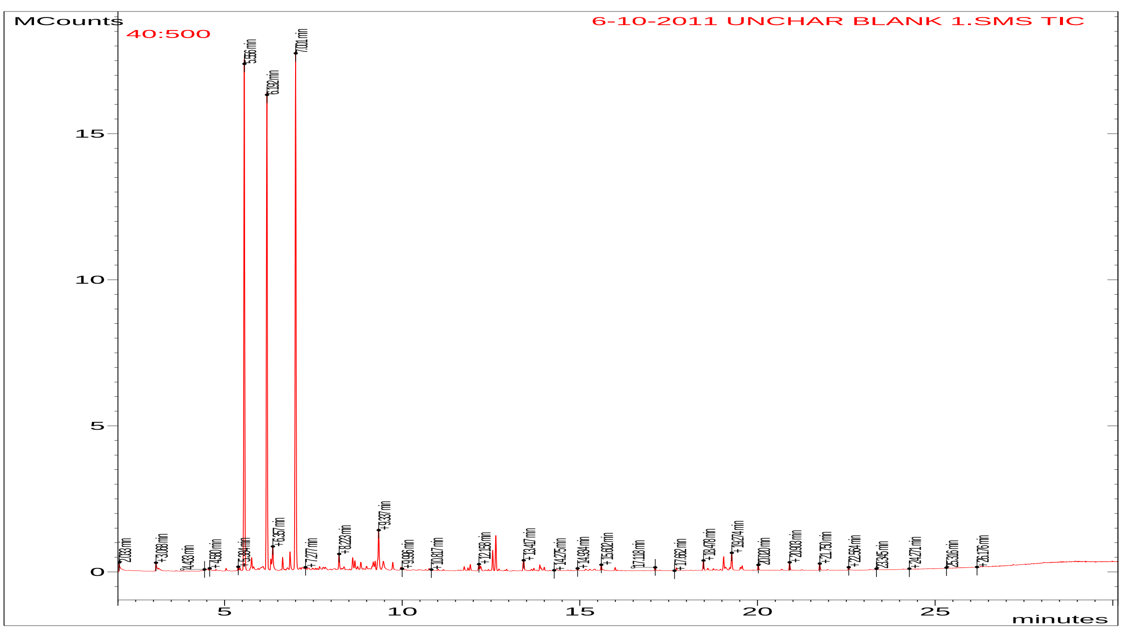


Figure 14: Total Ion Chromatogram (TIC) of Un-Charred Yellow Pine Sample with No Ignitable Liquid Spike

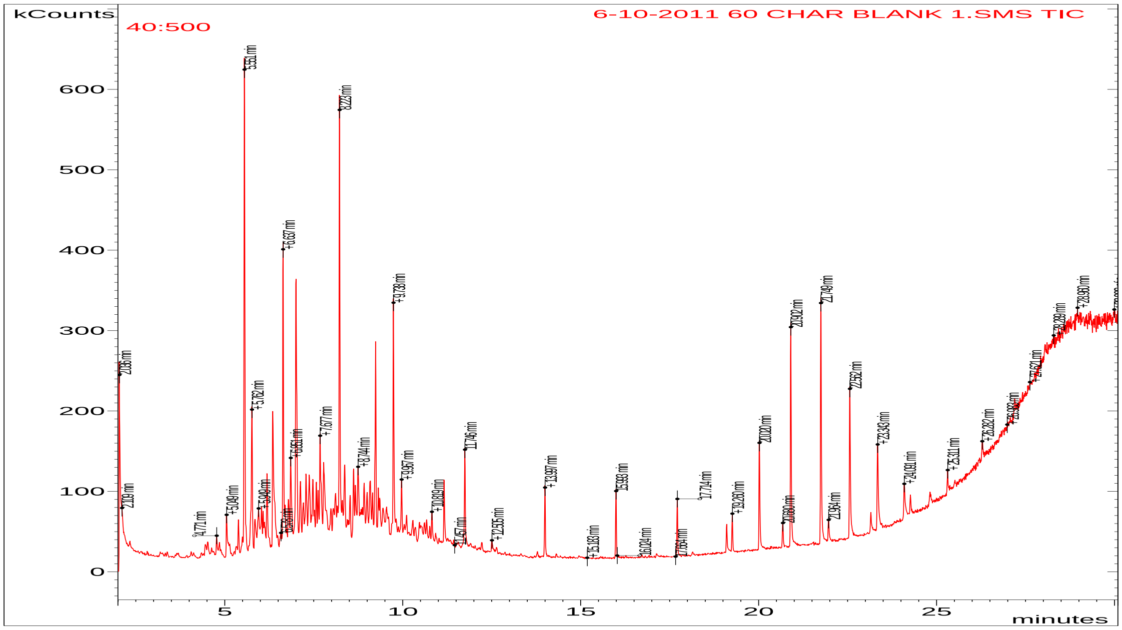


Figure 15: Total Ion Chromatogram (TIC) of Charred Yellow Pine Sample with No Ignitable Liquid Spike

For carpet, several minor precursory products were present in the substrate blanks, mostly olefins from the carpet fibers (such as dodecene, 8.6min [Figure 16]), however these were negligible in the gasoline (Figure 17), kerosene (Figure 19), and charcoal lighter fluid spiked samples (Figure 20).

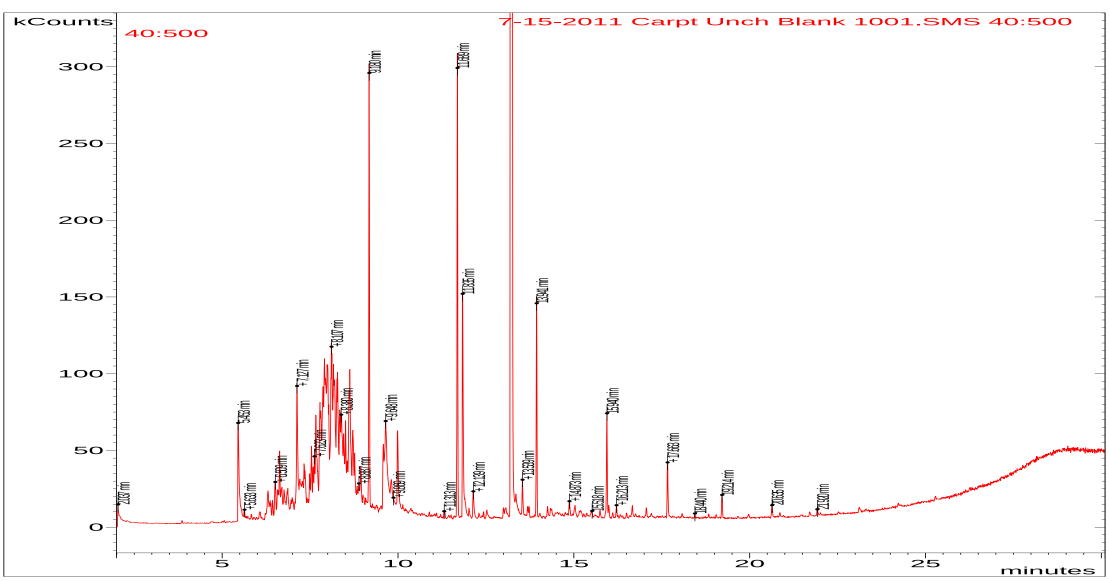


Figure 16: Total Ion Chromatogram (TIC) of Un-Charred Carpet Blank

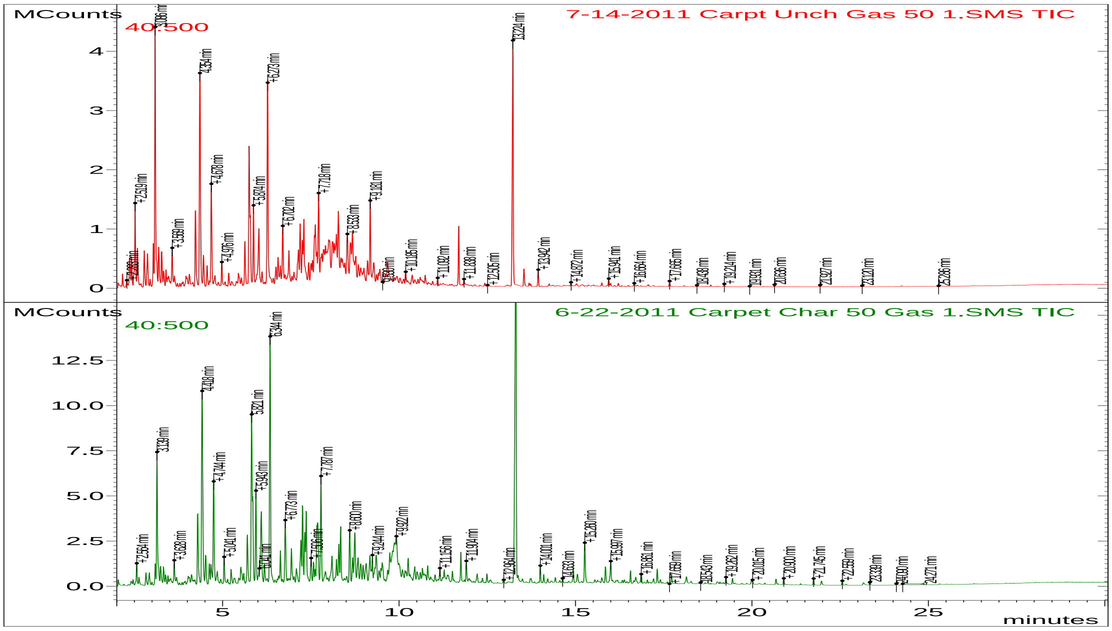


Figure 17: Total Ion Chromatograms (TICs) of Un-Charred and Charred Gasoline Spiked Carpet

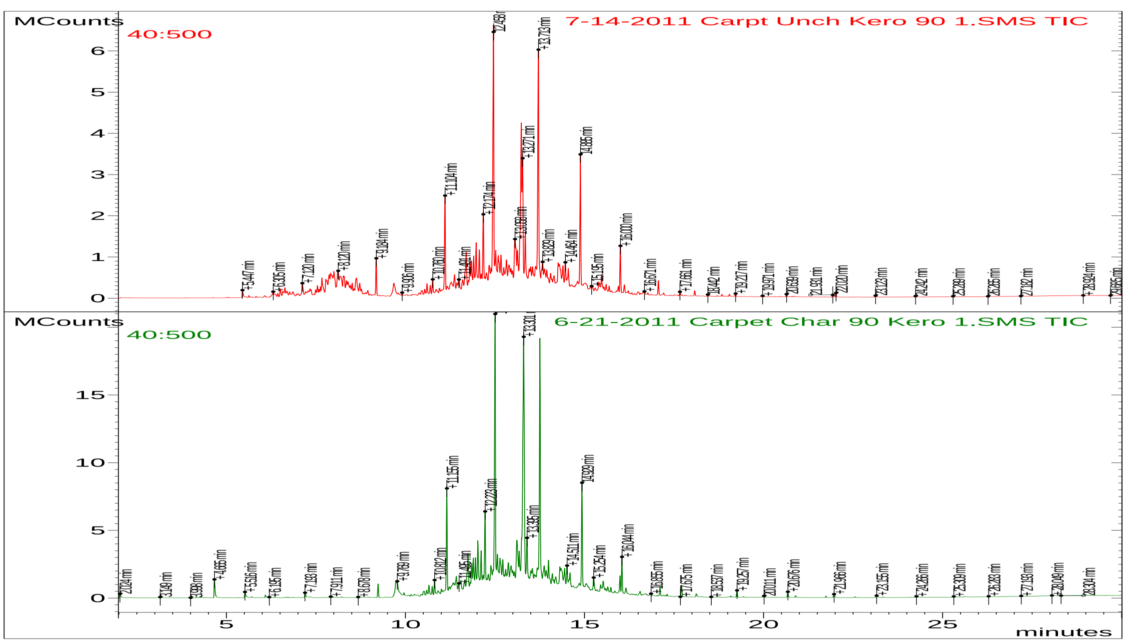


Figure 18: Total Ion Chromatograms (TICs) of Un-Charred and Charred Kerosene Spiked Carpet

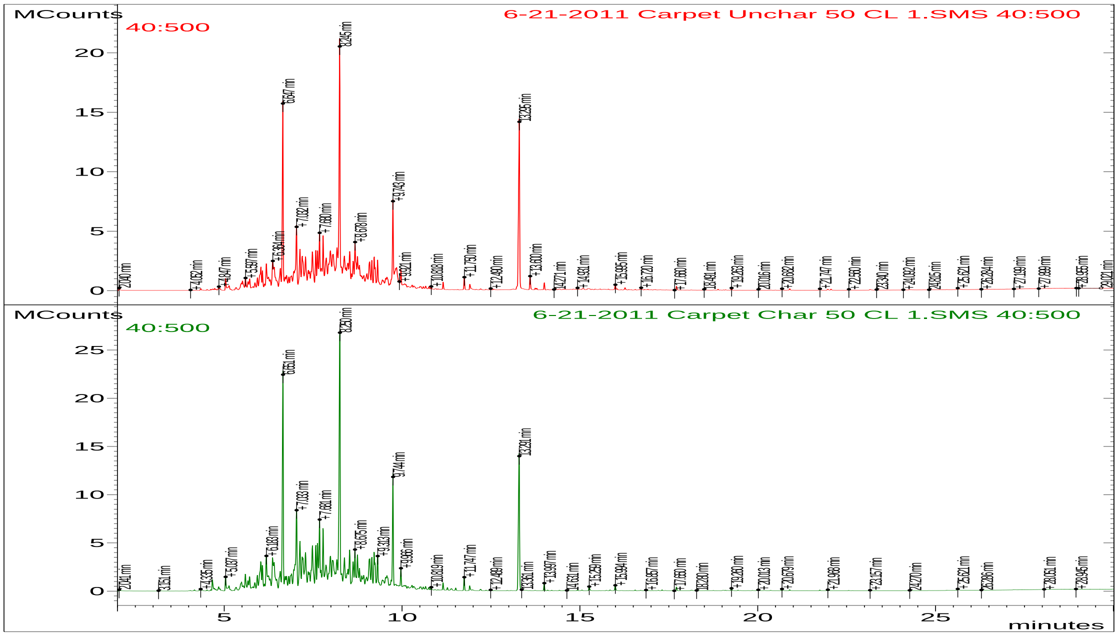


Figure 19: Total Ion Chromatograms (TICs) of Un-Charred and Charred Carpet Spiked with Charcoal Lighter Fluid

Carpet pad had very few detectable compounds in the substrate blanks, such as acenaphthene (11.9 min), a compound found in various dyes. (Figure 20). These compounds were also insignificant in the gasoline (Figure 21), kerosene (Figure 22), and charcoal lighter fluid spiked samples (Figure 23).

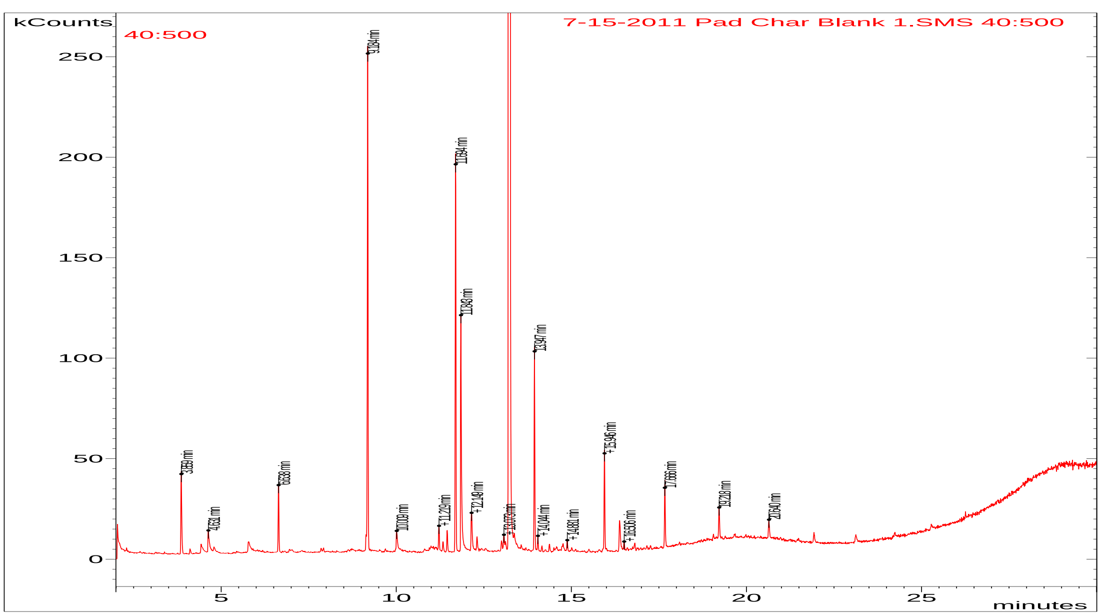


Figure 20: Total Ion Chromatogram (TIC) of Charred Carpet Pad Blank

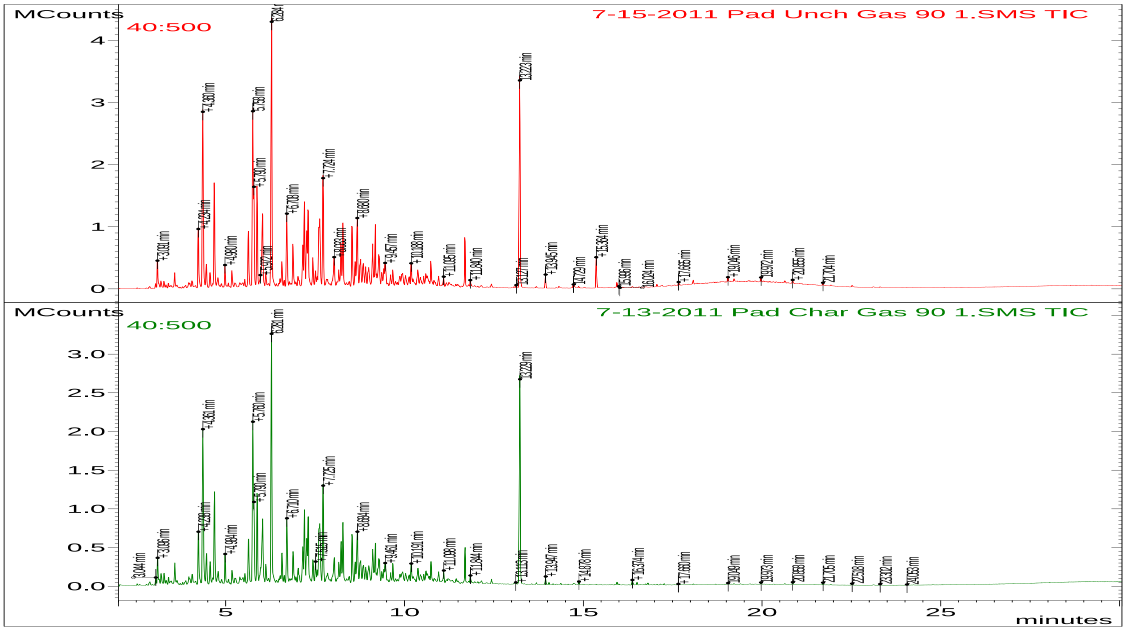


Figure 21: Total Ion Chromatograms (TICs) of Un-Charred and Charred Gasoline Spiked Carpet Pad

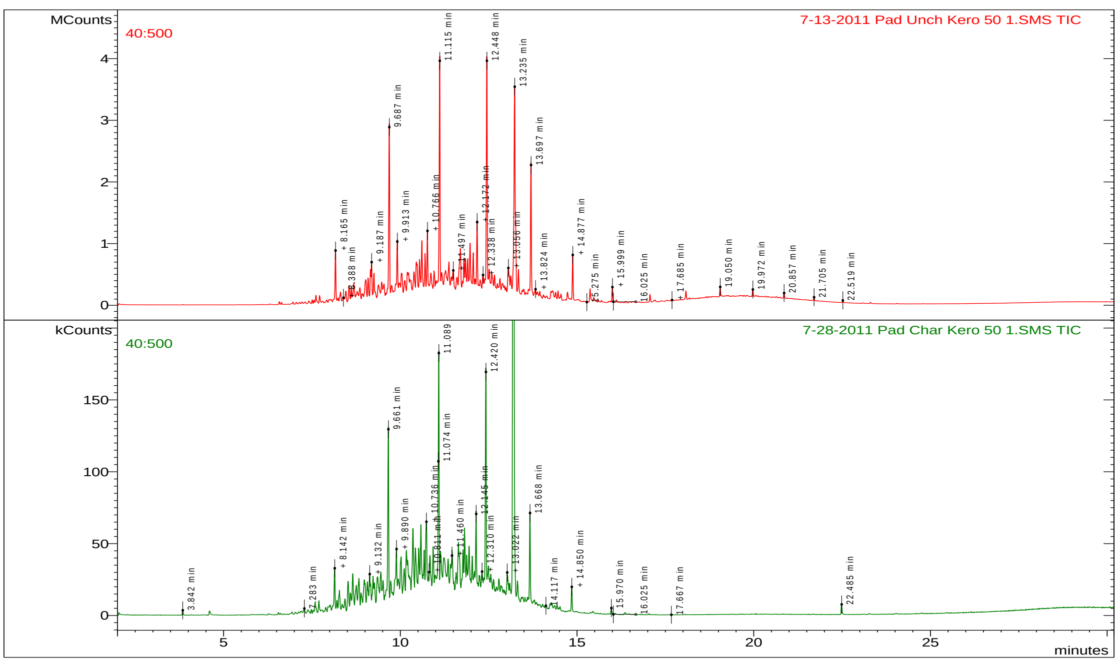


Figure 22: Total Ion Chromatograms (TICs) of Un-Charred and Charred Kerosene Spiked Carpet Pad

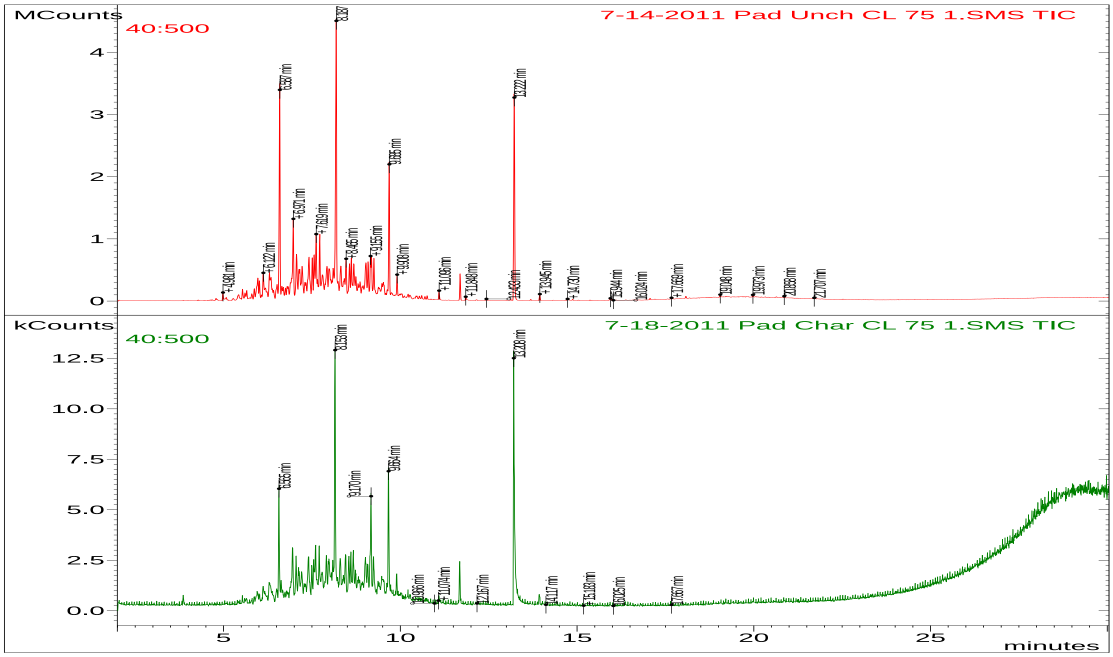


Figure 23: Total Ion Chromatograms (TICs) of Un-Charred and Charred Charcoal Lighter Fluid Spiked Carpet Pad

With kerosene the ignitable liquid pattern was found to shift to the lighter end of the chromatogram for both yellow pine and carpet pad, a phenomenon not observed with the other ignitable liquids or in the other substrates.

With yellow pine the entire chromatographic pattern was observed shifting approximately one carbon lower in comparison to the neat ignitable liquid (Figures 24 and 25). With all substrates various compounds were observed diminishing in abundance with charring. This was most pronounced in yellow pine where the relative abundance of the normal paraffins was significantly reduced relative to the branched and cyclic hydrocarbons. The reduced n-paraffins might lead to misclassifying a HPD as a naphthenic-paraffinic product [2]. For example, an ILR recovered from a container in possession by the suspect might be determined to be a HPD, whereas the ILR of that same HPD recovered from fire debris might be misclassified as a naphthenic-paraffinic product based on the greatly reduced n-hydrocarbon content due to selective retention. This might result in the analyst concluding the two samples were not related when in fact they were.

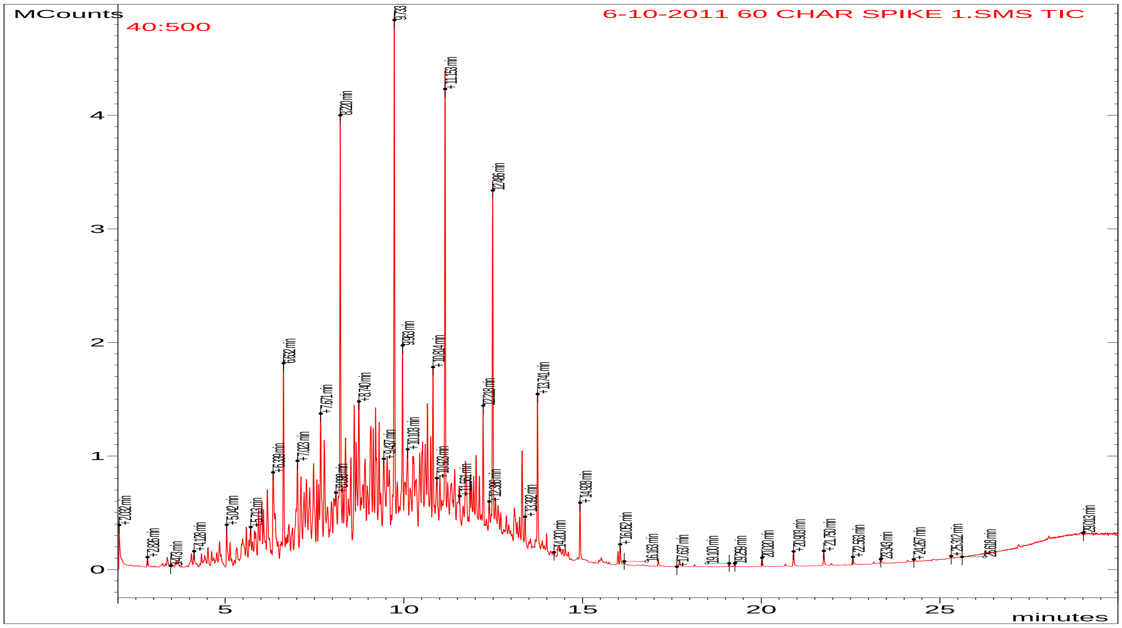


Figure 24: Total Ion Chromatogram (TIC) of Neat Kerosene

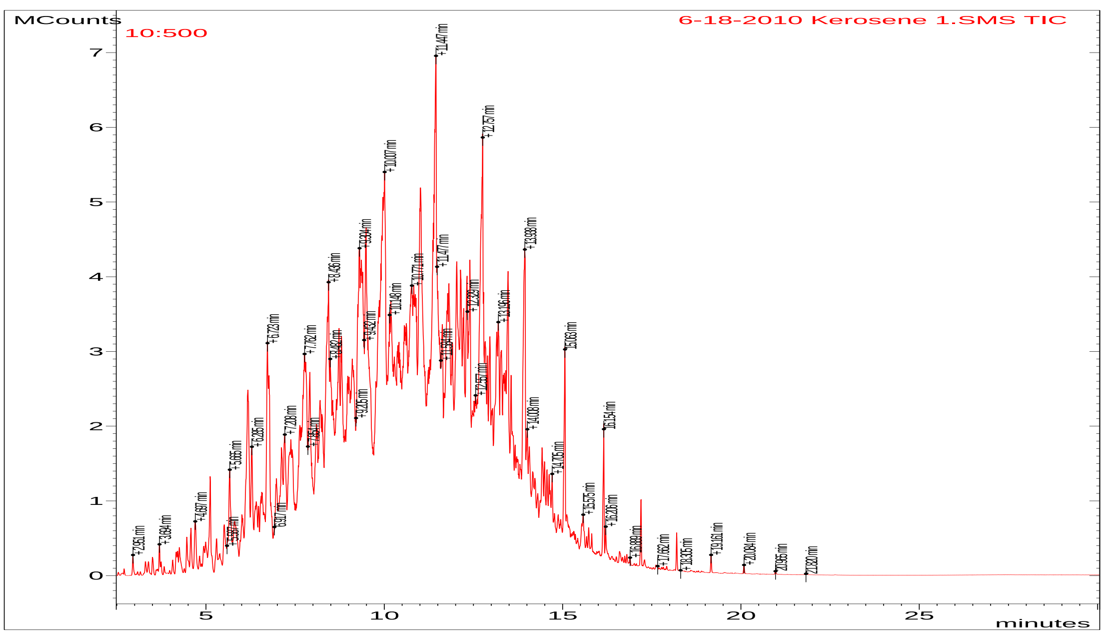


Figure 25: Total Ion Chromatogram (TIC) of Charred Yellow Pine Sample

The terpenes present in the un-charred yellow pine blanks were effectively buried in the samples that were un-charred and spiked with ignitable liquids. These products, therefore, did not affect the classification of the ignitable liquid. For carpet the precursory product peaks observed in the blank sample were insignificant in the ignitable liquid spiked samples, which also did not affect E1618 classification. In most samples the internal standard added (3PT, approximate retention time, 13.2 min) was the most abundant peak in the chromatogram. The few precursory product peaks present in minor concentrations in the carpet pad blank were easily masked in the baseline of the spiked samples and, therefore, were insignificant. The shifting of the chromatographic pattern observed in the charred yellow pine-kerosene spiked sample meant that the heavier components of the ignitable liquid were not recovered.

Several peaks (approximately 9.2 min, 11.7 min, 13.9 min, 15.9 min) appear with some regularity in the chromatograms of a variety of different samples. These peaks were identified from their mass spectra as various siloxane compounds which are utilized in the stationary phase of the gas chromatography column. The presence of these compounds in the generated chromatographs is a result, and indicator, of column degradation.

E1618 Expert System Methodology and Graphical User Interface

Using routines written in the software language R (included in the Appendix 1), each chromatogram was processed as a vector of discrete data [41]. A multivariate feature vector for each chromatogram has slots indicating the relative masses in each of the following extracted ion chromatograms: alkanes (m/z = 43, 57, 71, 85, 99), cycloalkanes (m/z = 55, 69, 82, 83), aromatics (m/z = 91, 92, 105, 106, 119, 120), alkylnaphthalenes (m/z = 128, 142, 156, 170), and indanes (m/z = 117, 118, 131, 132). The "peaks" (integrated areas) in the extracted ion chromatograms (EICs) were first scaled relative to the largest peak appearing across all the EICs. Relative mass (x) means the percentage of scaled mass accounted for in an EIC relative to the sum of scaled masses across all the EICs. The levels for these relative masses were **none** ($0\%<x\leq1\%$), **low** ($1\%<x\leq10\%$), **medium** ($10\%<x\leq30\%$), and **high** ($30\%<x\leq100\%$). Users of the R code can specify alternative mass cutoffs if desired.

The feature vector also contained binary descriptors (i.e. true/false) for the presence of an N-alkane pattern and hash-and-trash. Thus for example, chromatogram shown in Figure 26 below (TIC of an odorless, 1-K grade kerosene product):

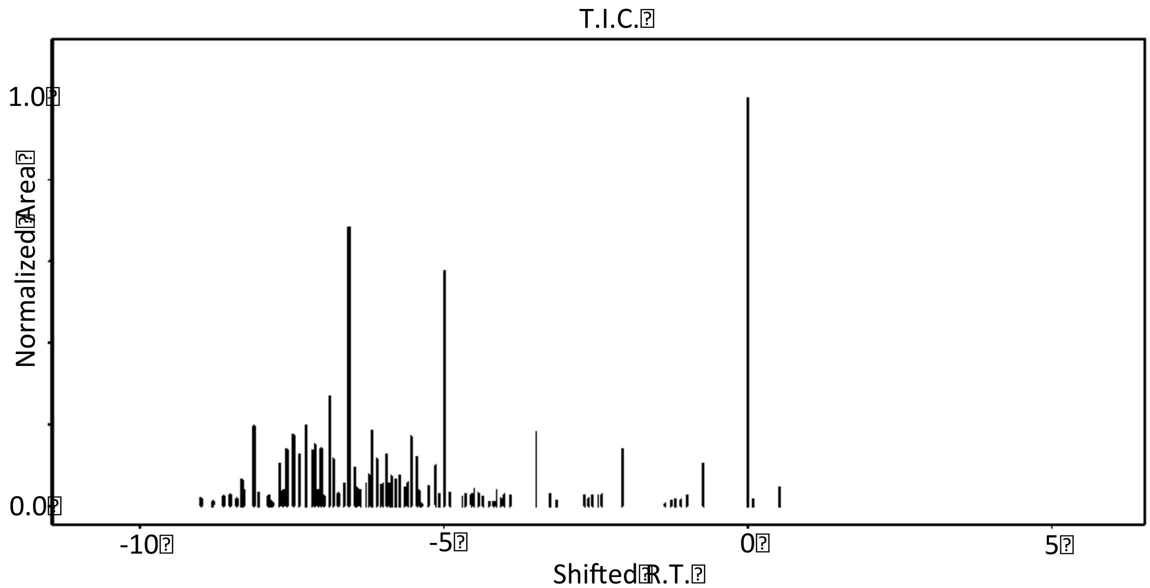


Figure 26 Total ion chromatogram for an odorless, K-1 grade kerosene product.

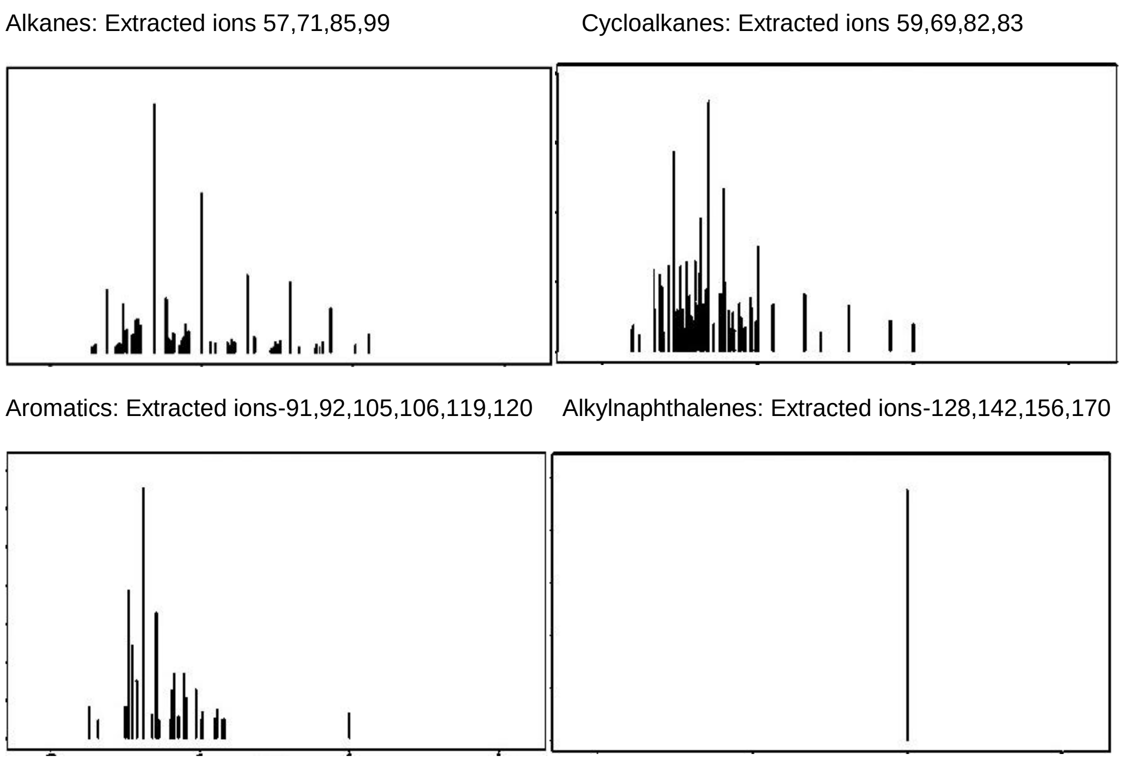


Figure 27 Select extracted ion chromatograms of TIC in Figure 26. Note, no indanes observed for the sample in Figure 26.and corresponding feature vector:

1. <u>N-alkane pattern?</u> = **True**,
2. <u>Hash-and-Trash?</u> = **True**,

3. Alkanes = **High**,
4. Cycloalkanes = **Medium**,
5. Aromatics = **Medium**,
6. Alkylnaphthalenes = **None**,
7. Indanes = **None**

Note that while the EIC for Alkylnaphthalenes shows a peak, it comprises <1% of the total scaled mass across all the EICs and so has level = **None**.

A total of 510 different ignitable liquid residues (ILR) were analyzed in this study and used to build an expert system. They are classified according to the E1618 standard as (light, medium heavy) gasolines, petroleum distillates (PD), isoparaffinic products, aromatics, naphthenic-paraffinic products (Nap-Par), normal-alkanes (N-alkane), oxygenated solvents and miscellaneous (Misc). Between four and six replicate runs of each ILR were performed to help the expert system cope with within-run variation. A 1633 by 8 raw case-list of all the ILR samples (1633) was assembled to train/test the system. Each row of the case list consisted of the seven-component feature vector shown above, along with the E1618 class designation. For example the first nine rows of the case list are shown in Table 6 below. It displays the data automatically extracted from the chromatograms of the first two ILRs.

Table 6 First nine rows of the raw case list automatically extracted form the raw chromatograms

Sample #	E1618 Class	N-alkane pattern*	Hash & Trash*	Alkanes	Cycloalkanes	Aromatics	Naphthalenes	Indanes
1	Misc	TRUE	TRUE	Med	Med	High	None	Low
2	Misc	TRUE	TRUE	Med	Med	High	None	Low
3	Misc	TRUE	TRUE	Med	Med	High	None	Low
4	Misc	TRUE	TRUE	Med	Med	High	None	Low
5	Misc	TRUE	TRUE	Med	Med	High	None	Low
6	PD	TRUE	TRUE	High	High	Med	None	None
7	PD	TRUE	TRUE	High	High	Med	None	None
8	PD	TRUE	TRUE	High	High	Med	None	None
9	PD	TRUE	TRUE	High	Med	Med	None	None

*N-Alkane pattern is the homologus series of normal alkanes predominate in petroleum distillates. Hash & Trash refers to the poorly resolved envelope of predominately iso-alkanes and cycloalkanes found in petroleum distillates.

One can think of each possible configuration of observable variables (N-alkane pattern, Hash&Trash, Alkanes, Cycloalkanes, Aromatics, Alkylnaphthalenes, Indanes) as probabilistically specifying any E1618 class. If the interdependences amongst these variables can be mined from a set of data, an approximation can be found of the joint probability mass function for the vector:

x = (Class, N-alkanes, Hash & Trash, Alkanes, Cycloalkanes, Aromatics, Alkylnaphthalenes, Indanes).

The joint probability mass function over the eight discrete variables characterizing an ILR can be compactly represented as an undirected graph known as a Markov random field. Such a graphical model can be used to specify Pr(**x**) and any of its marginals and conditionals. The conditional probability of most interest for this study was Pr(E1618 Class | **x**/{Class}), i.e. the probability of each E1618 class given the observed data (without class label) from the ILR EICs.

The R package gRim [42] was used to estimate pair-wise dependence between variables comprising **x** within the raw case-list and fit log-linear coefficients. Several Markov random fields were estimated using the Akaike information criterion (AIC), Bayesian information criterion (BIC) and our own domain specific expertise. After a Markov random field was fit, it was validated using hold-one-out cross validation (HOO-CV) to obtain an estimate of overall model error rate. The error was assessed to determine if the correct classification of the held out ILR was assigned a probability amongst the top three class probabilities produced. The graphical structure specifying some dependence between all the variables (i.e. the saturated model) preformed the best under of all the Markov random fields fit and is shown below.

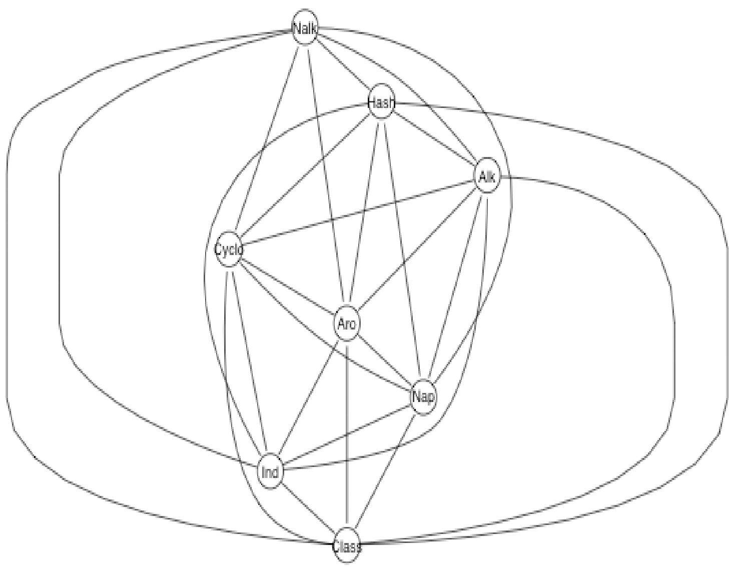


Figure 28: Representation of best performing Markov random field in terms of HOO-CV.

The table below lists the overall performance of the Markov random field expert system of Figure 3 under HOO-CV.

Table 7. Expert system performance under HOO-CV.

E1618 Class	Correct Class is in Top 3 Picks (%)
Gasoline	98.0
Light Petroleum Distillates (LPD)	97.1
Medium Petroleum Distillates (MPD)	99.7
Heavy Petroleum Distillates (HPD)	96.5
Light Isoparaffinic (LISO)	100
Medium Isoparaffinic (MISO)	100
Heavy Isoparaffinic (HISO)	88.0
Light Aromatic (LAROM)	93.8
Medium Aromatic (MAROM)	100
Heavy Aromatic (HAROM)	100
Light Naphthenic-Paraffinic[a]	-
Medium Naphthenic-Paraffinic (MNapPar)	100
Heavy Naphthenic-Paraffinic (HNapPar)	100
Light Normal-Alkane[a]	-
Medium Normal-Alkane[a]	-
Heavy Normal-Alkane (HNPar)	100
Light Oxygenated (LOXY)	96.8
Medium Oxygenated (MOXY)	96.9
Heavy Oxygenated HOXY)	88.6
Light Miscellaneous (LMisc)	100
Medium Miscellaneous (MMIsc)	95.8
Heavy Miscellaneous (HMisc)	88.5

[a]Not observed in the data set.

A graphical user interface (GUI) was developed to aid the fire debris examiner in their classification task for an unknown ILR. The GUI was built using the R packages RGtk2 and gWidgets [43,44]. A screen shot is shown in Figure 29 below.

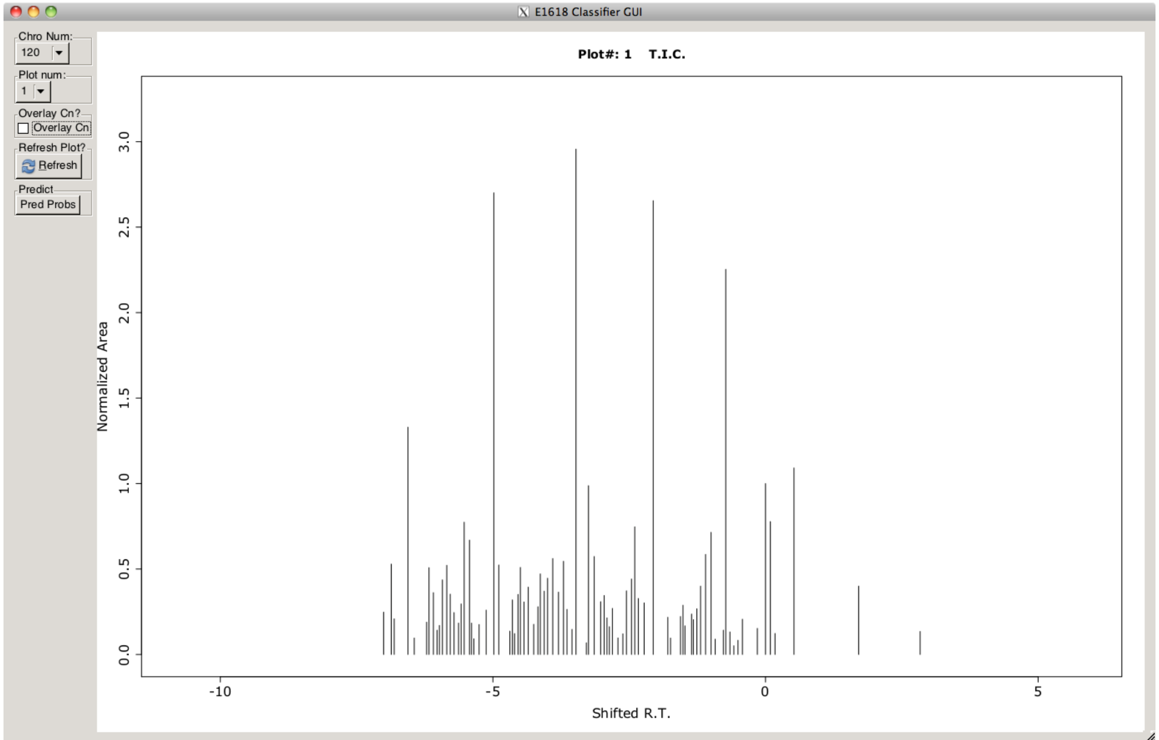
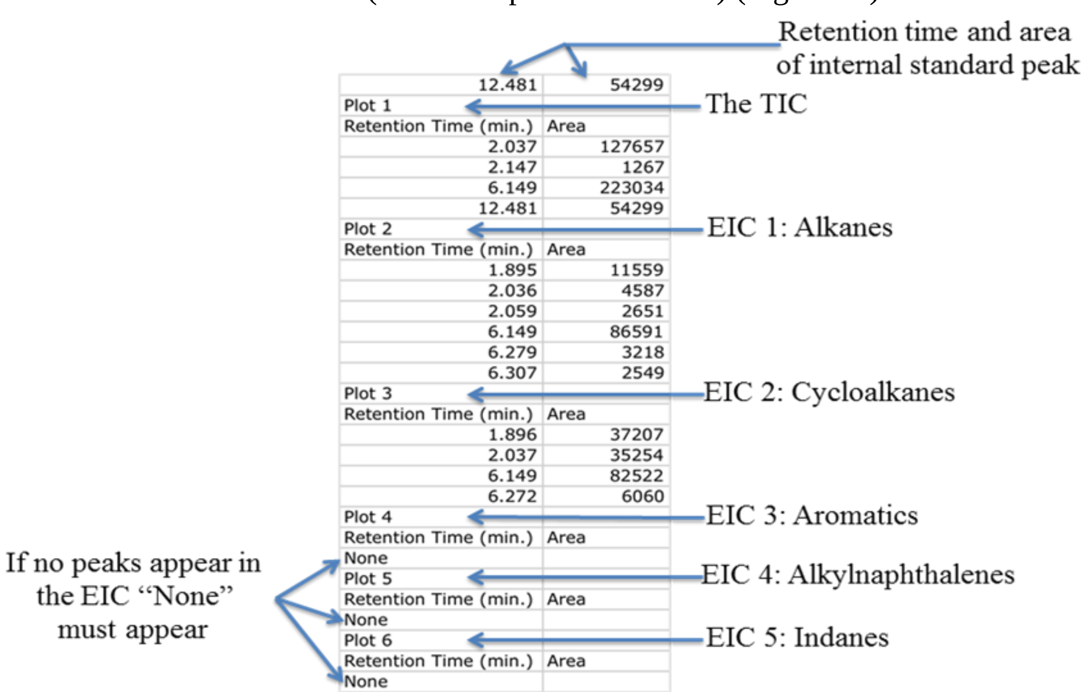


Figure 29: Screen shot of initial GUI window after opening an unknown TIC.

The GUI is used to open a TIC and EICs output in the following format from any instrument. It should be saved as a .csv file (comma-separated-variable) (Figure 30).



12.481	54299
Plot 1	
Retention Time (min.)	Area
2.037	127657
2.147	1267
6.149	223034
12.481	54299
Plot 2	
Retention Time (min.)	Area
1.895	11559
2.036	4587
2.059	2651
6.149	86591
6.279	3218
6.307	2549
Plot 3	
Retention Time (min.)	Area
1.896	37207
2.037	35254
6.149	82522
6.272	6060
Plot 4	
Retention Time (min.)	Area
None	
Plot 5	
Retention Time (min.)	Area
None	
Plot 6	
Retention Time (min.)	Area
None	

Figure 30: Required (textual) format for TIC/EICs for GUI compatibility.

Other fields output by specific instruments are permissible, as the GUI simply looks for the words "Plot", "Retention" and "Area" in the file. Note that "Retention" and "Area" fields must appear somewhere after each "Plot" field. Plot 1 is assumed to be the TIC. Plots 2-6 are the EICs and have the assumptions listed in Figure 30. If no peaks appear in an EIC the word "None" must appear in the Retention Time column. **Note that the first row of the file *must be* the retention time and area of the internal standard** (SPT).

Once the .csv file containing the TIC/EICs is opened the GUI automatically shifts the retention times to be with respect to the internal standard and scales the EIC peak heights (areas) to be with respect to the tallest peak across all the EICs. Note that the internal standard should not comprise one of the peaks of the EICs. With the GUI opened the TIC or any of the EICs can be viewed. Also the Cn (n = 6-20) can be superimposed with the check box if desired as shown below in Figure 31.

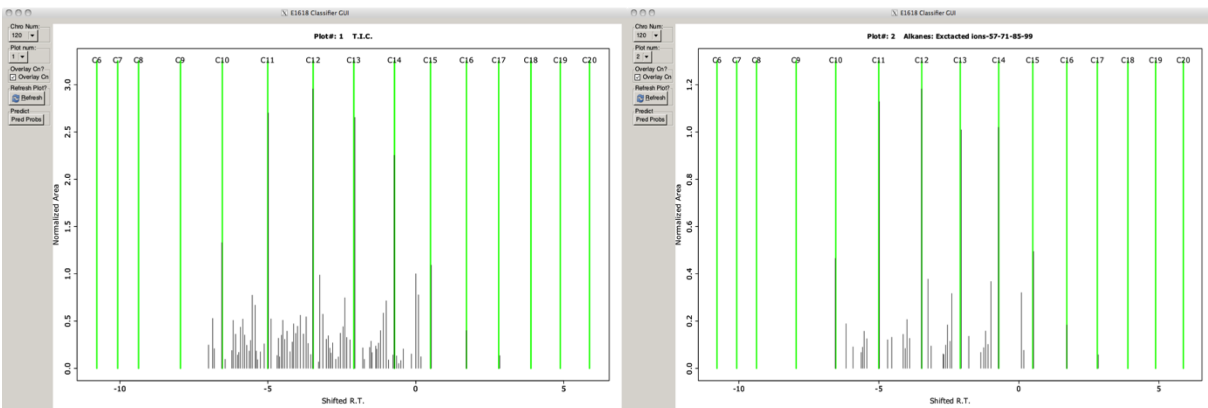


Figure 31: View TIC or any EIC. Superimpose Cn series if desired.

Finally, the fit Markov random field model can be queried using the R package gRain [45]. The feature vector is automatically extracted using data from the EICs when the "Pred Plot" button on the left hand side of the GUI is pushed. The feature vector instantiates the corresponding variables in the model and a probability is computed for each E 1618 Pr(E1618 Class | **x**/{Class}). The GUI displays the estimated probabilities as shown in Figure 32.

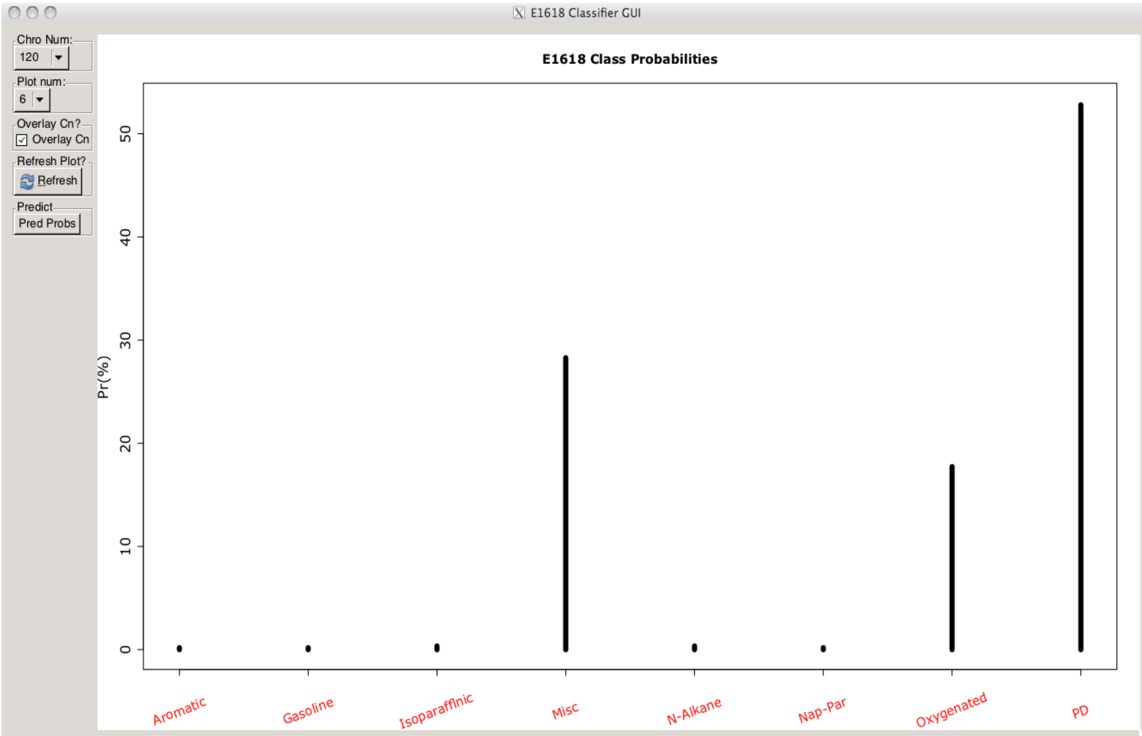


Figure 32: Posterior probability view for each E1618 class given the EIC data from an unknown ILR.

Challenging the Expert System with Spiked Substrate Samples

The complete results of the substrate test samples are given in the Appendix. A summary of the results of the challenge samples are given in Table 8.

Table 8: Challenge Samples Results

Substrate burned to 50% mass, then spiked with Ignitable Liquid				
File Name	**Ignitable Liquid**	**Class**	**Substrate**	**Expert System ID (top 3)**
Blc 54	Crown Paint Thinner	MPD	blue carpet	MOXY, MPD
Blc 59	Crown Lacquer	MOXY	blue carpet	Cannot ID

	Thinner			
Blc 66	Kingsford lighter fluid	MPD	blue carpet	MOXY, MPD
Blc Blk	unburned blank		blue carpet	Cannot ID
Blc E85	E85	Gasoline	blue carpet	MOXY, MPD
Blc Gas	Gasoline	Gasoline	blue carpet	MMisc
Blc Ker	Kerosene	HPD	blue carpet	HPD, HMisc, HOXY
Blc MtdB	burned blank		blue carpet	HMisc, HOXY
Brc 54	Crown Paint Thinner	MPD	brown carpet	MPD, MOXY
Brc 59	Crown Lacquer Thinner	MOXY	brown carpet	HOXY, Cannot ID
Brc 66	Kingsford lighter fluid	MPD	brown carpet	MOXY, MPD
Brc Blk	unburned blank		brown carpet	Cannot ID
Brc E85	E85	Gasoline	brown carpet	MMisc
Brc Gas	Gasoline	Gasoline	brown carpet	MMisc, MPD
Brc Ker	Kerosene	HPD	brown carpet	HPD, HMisc, HOXY
Brc MtdB	burned blank		brown carpet	Cannot ID
CP 54	Crown Paint Thinner	MPD	carpet pad	MNapPar, MPD, MOXY
CP 59	Crown Lacquer Thinner	OXY	carpet pad	Cannot ID
CP 66	Kingsford lighter fluid	MPD	carpet pad	MOXY, MPD
CP Blk	unburned blank		carpet pad	Cannot ID

CP E85	E85	Gasoline	carpet pad	MOXY
CP Gas	Gasoline	Gasoline	carpet pad	MMisc
CP Ker	Kerosene	HPD	carpet pad	HPD, HMisc, HOXY
CP MtdB	burned blank		carpet pad	Cannot ID
OO 54	Crown Paint Thinner	MPD	old oak	MNapPar, MOXY, MPD
OO 59	Crown Lacquer Thinner	OXY	old oak	HOXY
OO 66	Kingsford lighter fluid	MPD	old oak	MPD
OO Blk	unburned blank		old oak	Cannot ID
OO E85	E85	Gasoline	old oak	MMisc
OO Gas	Gasoline	Gasoline	old oak	MMisc, MOXY, MPD
OO Ker	Kerosene	HPD	old oak	HPD, HMisc, HOXY
OO MtdB	burned blank		old oak	HPD
PW 54	Crown Paint Thinner	MPD	Plywood	MPD, MOXY, MMisc
PW 59	Crown Lacquer Thinner	OXY	Plywood	HOXY
PW 66	Kingsford lighter fluid	MPD	Plywood	MMisc
PW Blk	unburned blank		Plywood	Cannot ID
PW E85	E85	Gasoline	Plywood	HXY
PW Gas	Gasoline	Gasoline	Plywood	MMisc
PW Ker	Kerosene	HPD	Plywood	HPD, HMisc, HOXY
PW MtdB	burned blank		Plywood	HOxy

YP 54	Crown Paint Thinner	MPD	Yellow Pine	MMisc, MPD
YP 59	Crown Lacquer Thinner	OXY	Yellow Pine	Cannot ID
YP 66	Kingsford lighter fluid	MPD	Yellow Pine	MMisc, MOXY
YP Blk	unburned blank		Yellow Pine	MMisc, Gasoline
YP E85	E85	Gasoline	Yellow Pine	MMisc, MOXY
YP Gas	Gasoline	Gasoline	Yellow Pine	MMisc
YP Ker	Kerosene	HPD	Yellow Pine	HPD
YP MtdB	burned blank		Yellow Pine	Cannot ID
	Substrate spiked with IL then ignited			
GasYP	Gasoline	Gasoline	Yellow Pine	MMisc
KerYP	Kerosene	HPD	Yellow Pine	Cannot ID
	IL spiked on unburned substrate			
UBSBLC54	Crown Paint Thinner	MPD	Unburned blue carpet	MNapPar, MOXY, MPD
UBSBLC59	Crown Lacquer Thinner	MOxy	Unburned blue carpet	Cannot ID
UBSYP54	Crown Paint Thinner	MPD	Unburned Yellow Pine	MOXY, MPD, MMisc

Table 9 summarizes the errors for the challenge samples. The percent correct below is the number correctly identified by class rather than overall.

Table 9. Summary of Correct E1618 Classification of Challenge Ignitable Liquid Residues

Ignitable Liquid	Correct Assignment	Percent Correct by Expert System (in top 3)
Crown Paint Thinner	MPD	100% (8/8)
Crown Lacquer Thinner	MOXY	0% (0/6) (4 not ID)
Kingsford Lighter Fluid	MPD	29& (2/7)
E85 Gasoline	Gasoline	0% (0/6)
87 Octane Gasoline	Gasoline	0% (0/7)
Kerosene	HPD	86% (6/7) (one not ID)
Blanks	"Cannot ID"	64% (7/11)

Although the results of the expert system with the ILRC data were quite acceptable, the challenge samples on spiked substrates were disappointing. The same set of ignitable liquids (a paint thinner, lacquer thinner, charcoal lighter, E85 gasoline, 87 Octane gasoline and kerosene) were applied to burnt substrates then processed in metal paint cans as per E1412 method. Both gasoline samples were consistently misidentified. This is most vexing as gasoline is a common accelerant used in arson cases. It could be argued that E85 gasoline is really an oxygenated solvent because the majority of the product is ethanol, however, ethanol being very water soluble, it is likely to be removed during fire suppression and not observed. Kerosene (HPD) and paint thinner (MPD) were generally correctly classified in the top three picks by the expert system. The lacquer thinner (medium oxygenate) was either mis-identified as a Heavy Oxygenate (HOXY) or not able to be identified (Cannot ID). Also of concern were the occasional blank sample given a E1618 classification (Table 9). This is considered a False Positive and a serious error. Raising the limit of relative peak area in the TIC for classification would most likely remove this problem. Not being able to classify a sample when an ignitable liquid is present can be considered a False Negative, however, in an actual case, this is not as onerous as a false positive. Additional ignitable liquids should be analyzed including the new substrate collection in the ILRC to help resolve the errors observed here.

Determination of Error Rates by Fire Debris Analysts

As discussed in the methods section above, false negatives ("no ILR detected" when one was present) and inability to make an classification when ILR was detected, were not considered errors as they follow the premise that the the accused is innocent until proven guilty. No false positive or mis-classification errors were observed. A number of samples which were positive were not classified by the participants based on protocols in their respective labs (lack of comparison samples, low levels, etc.). As the participants were aware of the nature of the study, there may have been some “conservative” bias, that is, to be reluctant to make a determination because “no classification possible” was not considered an error whereas mis-classification would be an error. Further studies with a larger group including federal laboratories are highly recommended.

Ignitable Liquid Pour Patterns

For each set of analyses on the GCMS an E-1618-97 standard test mixture (Restek Co., Bellefonte, Pennsylvania) was prepared and analyzed. From this standard, important peaks could be identified in the chromatogram. A ratio of the peak area of tetradecane for the kerosene samples and 1,2,4-trimethylbenzene for the gasoline samples to the peak area of the 3-phenyltoluene internal standard was calculated to normalize the chromatographic data to the amount of IL residues present. A higher percentage indicated a greater amount of IL residues present. Since each sample was run in triplicate, the average ratio was calculated for each one.

A graph displaying the relative IL residues present in each sample in relationship to the location of removal on the substrate for the low pile carpet with kerosene pour in a circular pattern can be seen in Figure 32.

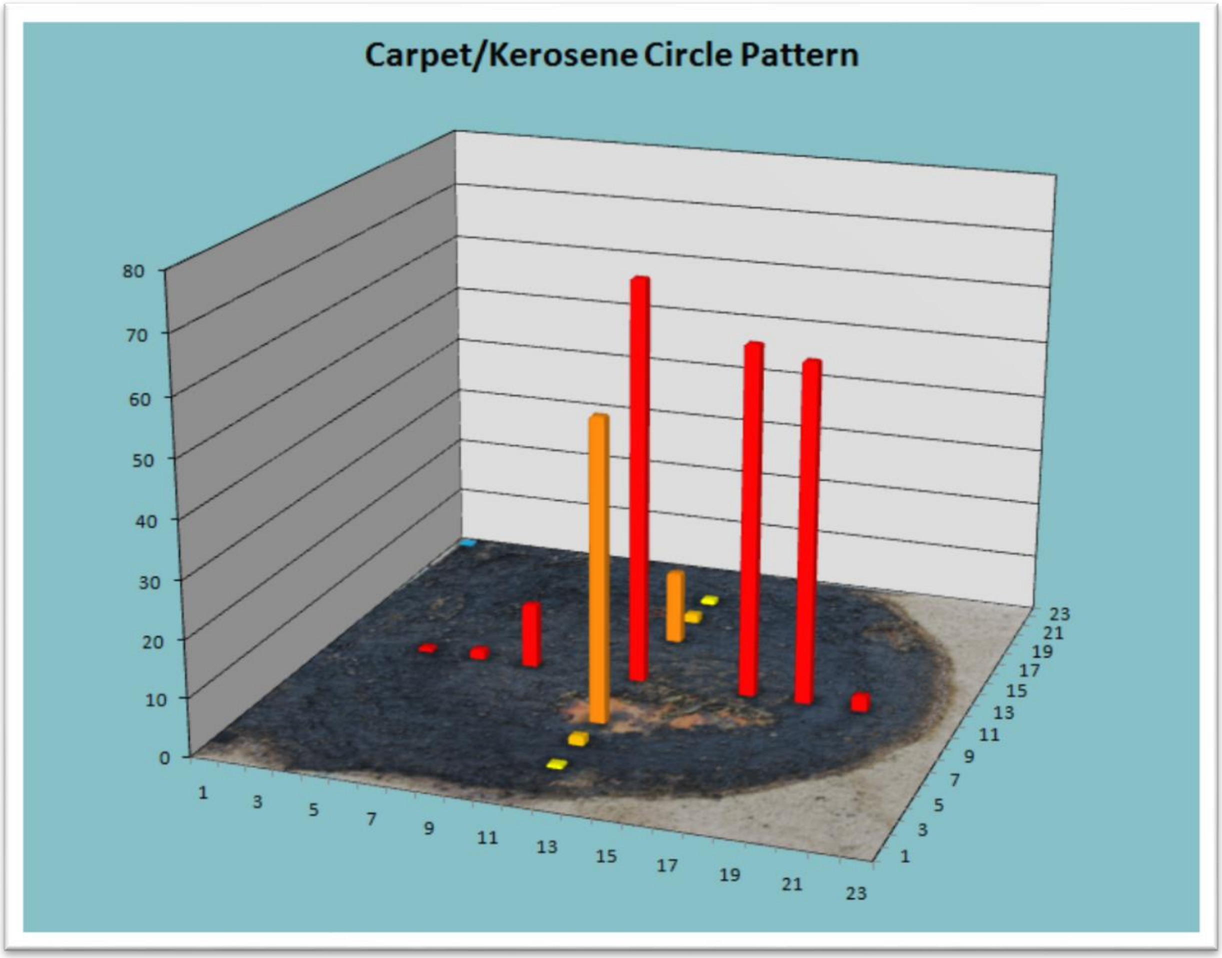


Figure 32: Graph of the percent ratios of the peak areas of tetradecane to 3-phenyltoluene (IS) for each sample in relation to the place each was sampled from on the carpet square.

Figure 33 shows the relative IL residues for gasoline on carpet in a circular pour. Figure 34 is of kerosene residues from a linear pour patterns. The photograph at the bottom of each figure shows the extent of the burn.

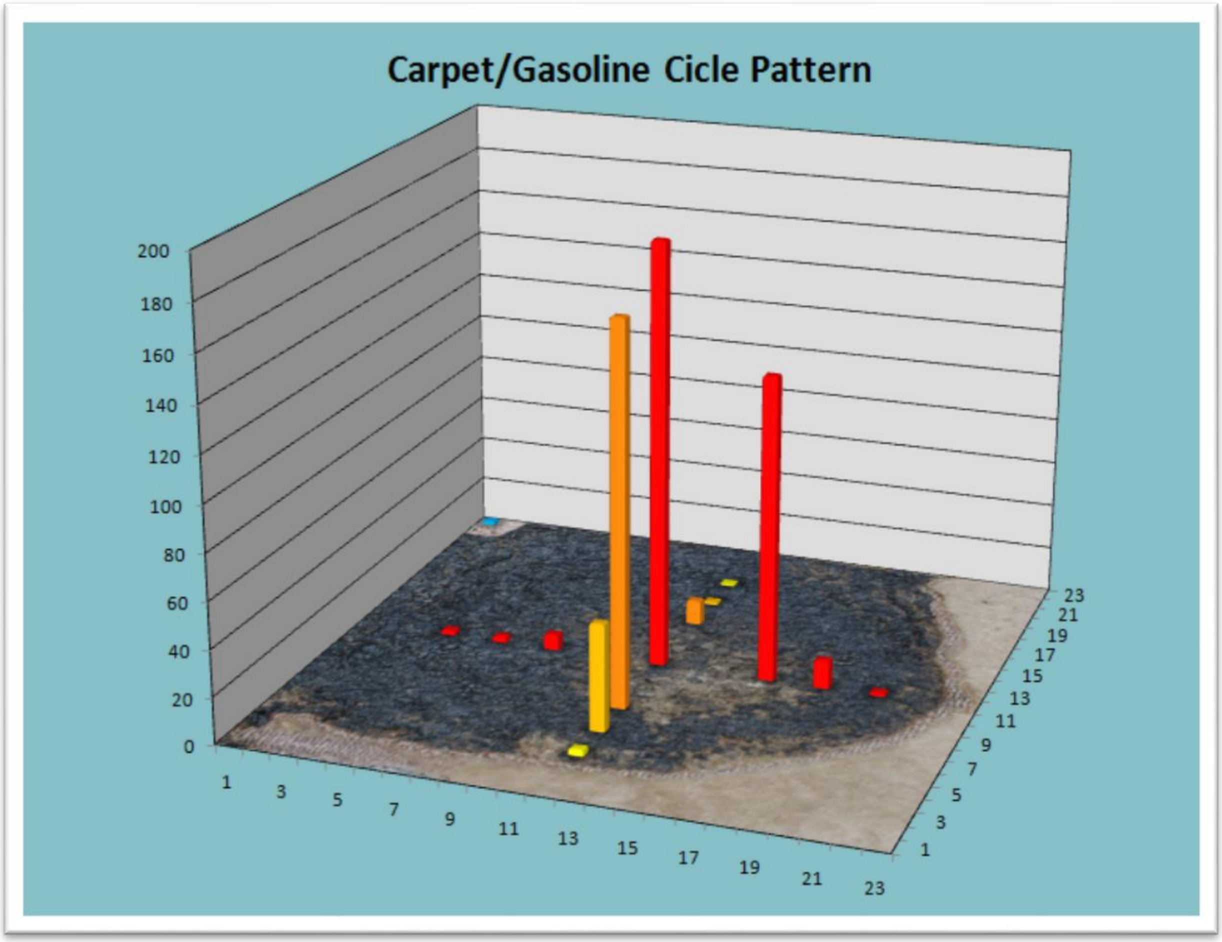


Figure 33: Graph of the percent ratios of the peak areas of 1,2,4-trimethylbenzene to 3-phenyltoluene (IS) for each sample in relation to the place each was sampled from on the carpet square. Photo (bottom) shows extent of burn.

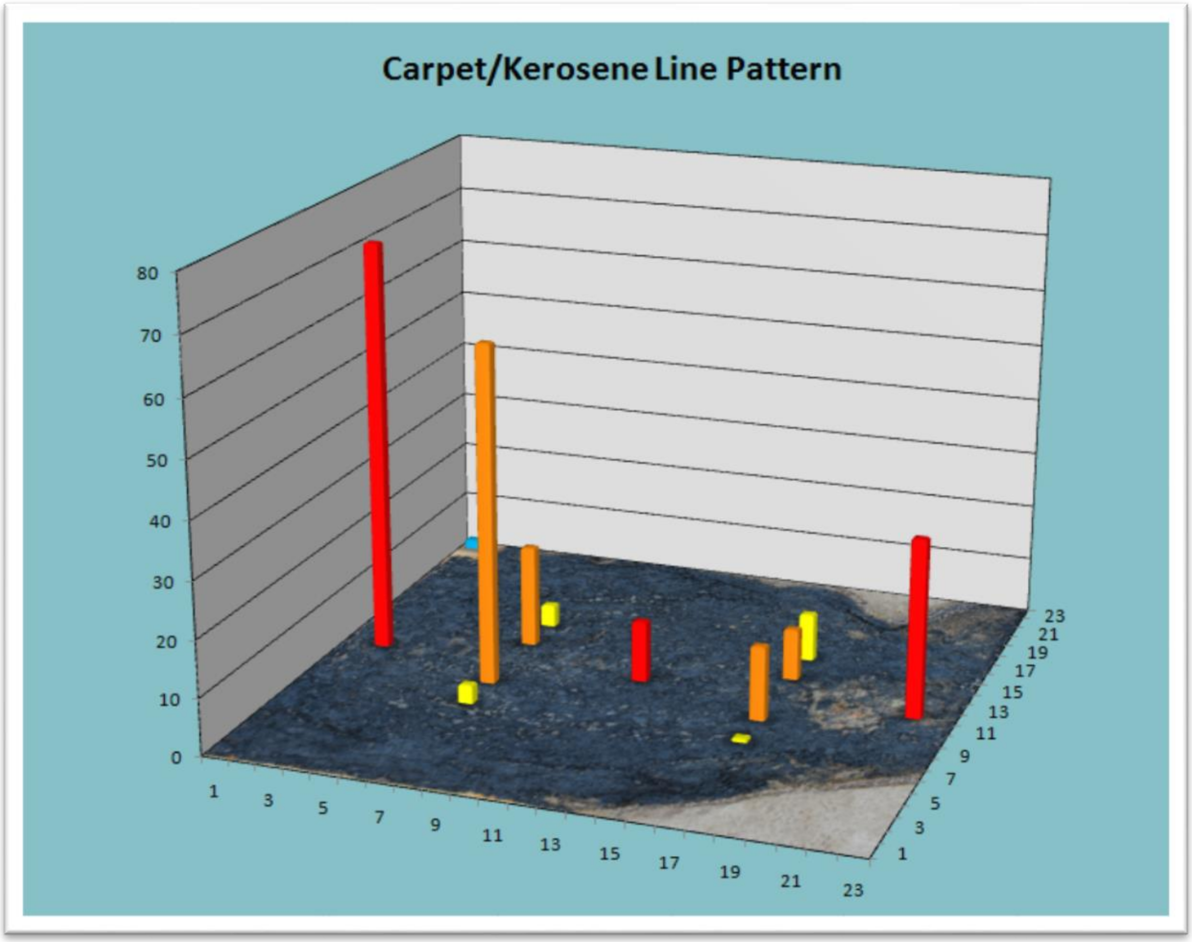


Figure 34: Graph of the percent ratios of the peak areas of tetradecane to 3-phenyltoluene (IS) for each sample in relation to the place each was sampled from on the carpet square.

The highest levels are towards the ends and along the axis of the pour. Because the ignitable liquid was pour by hand along the axis, it is likely that there was variation in the amount of liquid initially at any point. This may have resulted in some of the variation of recovered ILR along the axis. The amount off-axis does decrease rapidly as expected.

Oriented strand board (OSB) is a common subflooring material in home construction [32]. It is less absorbent than carpet so it was tested as an alternative material. Figure 35 the burn pattern for a circular pour pattern from 50 ml of kerosene on the OSB panel.

Figure 35: Burn pattern for OSB (50 ml of kerosene)

A significant amount of tetradecane (~6% relative to IS) was observed in the chromatogram of the control sample (X in Figure 2 above), this was subtracted from each value and replotted as shown in Figure 36. It can be seen that that the center sample was lower for the OSB than with the carpet (Figure 32) but still higher than towards the edges. It was observed during the pour experiments that the liquid tended to spread out more on the OSB probably due to its lower absorbency than with carpet.

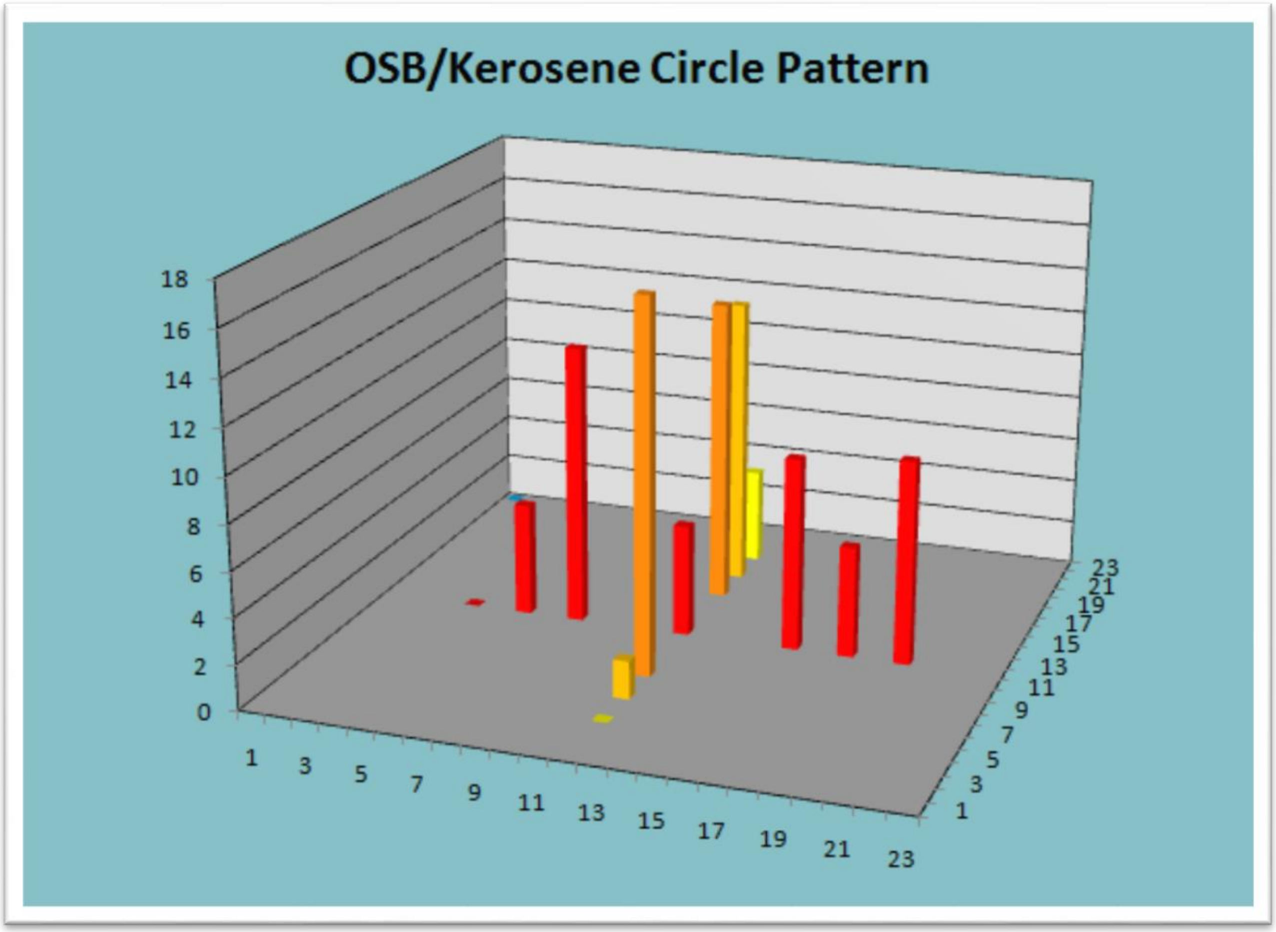


Figure 36: Graph of the percent ratios of the peak areas of tetradecane to 3-phenyltoluene (IS) for each sample minus 6.86% to compensate for the amount of tetradecane found in the blank OSB sample in relation to the place each was sampled from on the OSB square.

Two large scale test burns were conducted in the two bedrooms of a house being burned for a training exercise for a local fire department in West Virginia. Gasoline was used in one room (Room 1) and diesel in the other (Room 2), both poured in an S pattern. Samples were collected in a variety of locations in Room 1 and straight across the pour pattern in the room in Room 2 (Figure 37). In each case the highest results were seen nearest to the original pour pattern with lower amounts of IL residues being found further away. Orange tape was laid along the pour pattern after suppression of the fire to highlight where the pour was in the photograph.

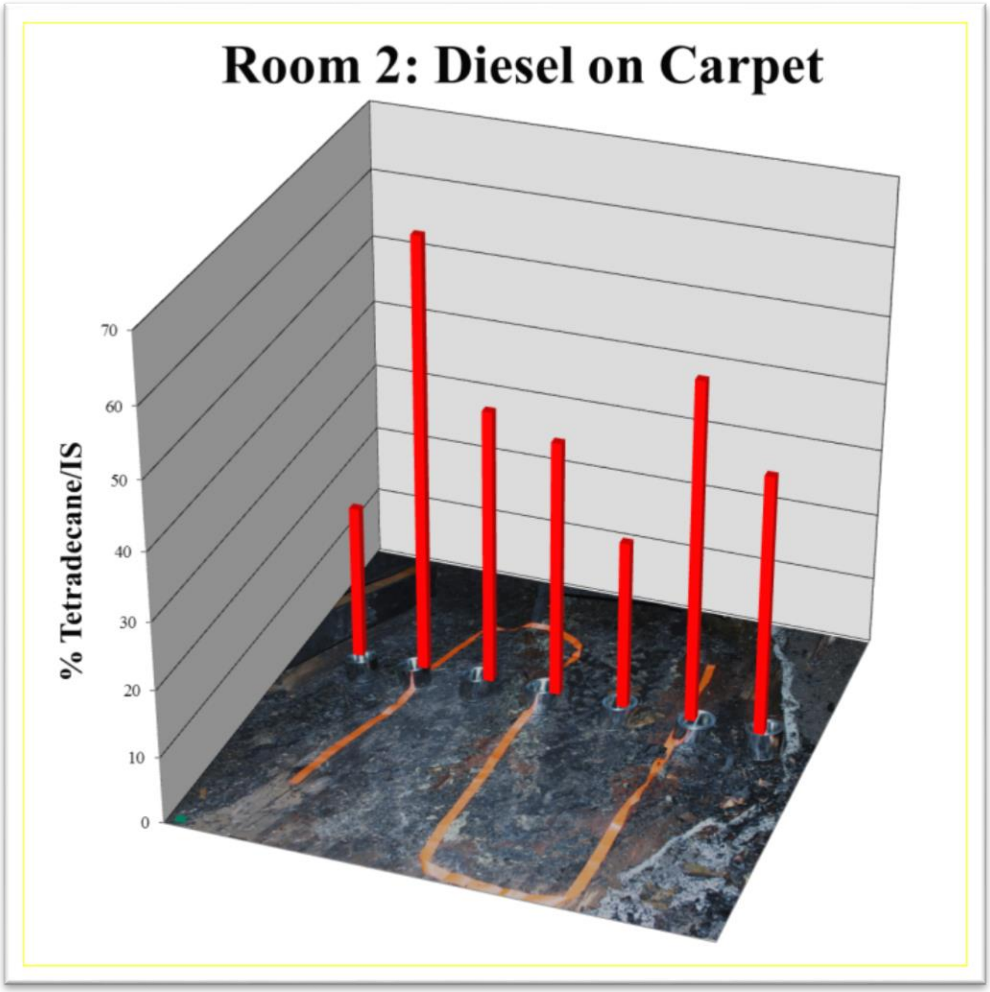


Figure 37. Graph of the percent ratios of the peak areas of tetradecane to 3-phenyltoluene (IS) in relation to the place each was sampled from in the room. Photo (bottom) shows extent of burn, shape of initial pour, and the location of each sample.

CONCLUSIONS

Substrate Effects

While previous observations that selective retention of certain compounds by the fire debris and pyrolysis products from the substrate may affect the assignment of the ILR to the correct E1618 class, this additional work confirms those observations with a range of substrates tested. While virtually all precursory products identified had little to no effect on analysis, these products were present to some degree in the ignitable liquid analysis and, under extremely low ignitable liquid concentrations, could appear much more pronounced. With the kerosene spiked and charred yellow pine the abundant precursory product terpenes virtually disappeared during burning (a phenomenon reasonably explained by the volatility of terpenes). However, in a substrate charred to a much lesser percentage, these products may be much more prominent in the analysis chromatogram and could affect classification. The shifting of the chromatographic pattern observed in the kerosene spiked and charred yellow pine did not change the classification of kerosene as a heavy petroleum distillate (C_8-C_{20+}), but the loss of the heavier components in the analysis under the right circumstances (such as low ignitable liquid concentration) may lead to misclassification. Additionally, the reduction in the relative abundance of the normal paraffins could potentially result in misinterpretation of the analytical results and misclassification of the ignitable liquid due to the fact that presence and abundance of certain compounds are a part of the classification criteria. Therefore, analysts should remain cognizant of the substrate being analyzed as it may affect their interpretation and E1618 classification of the results.

Expert System

The expert system on cross validation correctly predicted the E1618 class within the top three picks with better than 95% accuracy for samples taken from the ignitable liquid collection maintained by NCFS. However, the substrate challenge was less successful especially for gasoline samples which are the most common ILR found in arson cases. Therefore, refinements to the expert system are needed before "beta testing" in fire debris analysis labs. Additional samples not included in the training set and spiked substrate samples are necessary.

Error Rate Study of Fire Debris Analysts

No false positives were reported for the small sample size tested during this project. Additional larger studies would be necessary before an error rate can be determined.

Ignitable Liquid Pour Patterns

Overall, the variety of experiments performed thus far have shown that sampling closer to the center of a pour pattern should allow for the best obtained chromatographic results at the lab. Significantly higher amounts of ILRs were found at the center of the patterns compared to the outer edges. Also, the IL residues that are present toward the edges of the pattern are much more weathered with more of the lighter end compounds and distinguishable peaks lost. Sampling closer to the center appears to give chromatograms more closely related to those seen in unweathered IL samples.

Implications for policy and practice

The substrate effects confirm previous studies showing that competitive adsorption of ignitable liquid residues by charred substrates can affect the pattern and relative abundance of individual components in the residues. This could lead to misclassification of ILR according to E1618. Fire debris analysts should be aware of this possibility especially in residential structures where yellow pine is a common building material.

The related pour pattern study contradicts the "conventional wisdom" of sampling near the edge of a pour pattern. Fire investigators are encouraged to sample closer to the apparent center of the pattern and even across the pattern to increase the likelihood of sufficient ILR for identification and classification.

The expert system needs additional development before it can be recommended for incorporation into routine case work. One benefit of the approach undertaken here is that the software developed is open-source and independent of the vendor GCMS in use at a particular lab.

With additional studies to determine the accuracy of experienced fire debris analysts in identifying and classifying ILR especially at low levels, a reasonably average error rate for false positives and misclassification can be achieved. This will be of substantial benefit where such error rates are expected by the court.

Implications for further research

Further studies with larger participation of experienced fire debris analysts are needed to better determine the error rates and misclassification at low error rates. Based on this study there is an expectation that false positive rates will be very low as it appears that protocols from many agencies are conservative in the determination of the presence of an ignitable liquid residue. Misclassification may also be rare as well especially at lower levels as most laboratories seem to have protocols in place to prevent "over interpretation" in those cases. A more extensive set of "case files" should be developed with duplicate sets with and without additional "information" (i.e. "canine hit") to determine if these lead to a bias in favor of making a determination for low level samples.

The expert system developed here is in need of further refinement and testing before dissemination to agencies for incorporation in fire debris analysis. Inclusion of more samples from the ILRC as well as the NCFS substrate datebase and testing with more challenge samples not in the training set will be needed to validate the resulting system.

ACKNOWLEDGEMENTS

The authors wish to acknowledge the efforts of graduate students at Marshall University who contributed many long hours in collecting the data presented here especially, Amanda Heeren, Jared Vititoe, Kendra Wilbur, David Eckre, Kelsey Rusbarsky, and Katelyn Timmons. Special acknowledgement is given to Marshall Graduate Student, Dana Greely and Cabell County Fire Investigator, Bob Sullivan for the excellent study on pour patterns included in this project.

The National Institute of Justice and Marshall University are acknowledged for funding this research and support of the graduate students listed above.

The reviewer comments and recommendations are greatly appreciated.

REFERENCES

1. Williams MR, Fernandes D, Bridge C, Dorrien D, Elliott S, Sigman M. Adsorption saturation and chromatographic distortion effects on passive headspace sampling with activated charcoal in fire debris analysis. J. Forensic Sci. 2005; 50 (2) 1–10.
2. Kelly, RL. Competitive adsorption of ignitable liquids on charred wood. Proc. Am. Acad. For. Sci. 2010; Abstract A91.
3. Bondra AM, Trader CE, Rankin JG. The differentiation of kerosene samples by target compound ratio analysis. Proc. Am. Acad. For. Sci. 2010; Abstract A103.
4. Rankin JG, Heeren A, "Effects of Competitive Adsorption on Interpretation of Ignitable Liquid Residues in Fire Debris Analysis" Proc. Am. Acad. For. Sci. 2011, Abstract A134.
5. ASTM E1618-11 Standard Test Method for Ignitable Liquid Residues in Extracts from Fire Debris Samples by Gas Chromatography-Mass Spectrometry, ASTM Annual Book of Standard, ASTN International, West Conshohocken, PA, 2011.
6. Rankin JG, Greely DL, Sullivan B, "Ignitable liquid residue distribution in pour patterns". Proceedings of ISFI2012, Baltimore, MD October 2012.
7. Holzer G, Bertsch W, Zhang QW. Design criteria of a gas chromatography-mass spectrometry based expert system for arson analysis. Analyt. Chim. Acta 1992; 259 (2): 225-235.
8. Tan B, Hardy JK, Snavely RE. Accelerant classification by gas chromatography/mass spectrometry and multivariate pattern recognition. Analyt. Chim. Acta 2000; 422 (1): 37-46.
9. Sigman ME, Williams MR, Covariance Mapping in the Analysis of Ignitable Liquids by Gas Chromatography/Mass Spectrometry. Anal. Chem. 2006, 78 (5), 1713-1718
10. Bodle ES, Hardy JK. Multivariate pattern recognition of petroleum-based accelerants by solid-phase microextraction gas chromatography with flame ionization detection. Analyt. Chim. Acta 2007; 589: 247–254.
11. Sigman ME, Williams MR, Castelbuono JA, Colca JG, Clark CD. Ignitable Liquid Classification and Identification Using the Summed-Ion Mass Spectrum. Instrument Sci & Tech., 2008; 36 (4): 375 – 393.

12. Wan N, Desa SM, Nic Daéid N, Ismail D. Savage K . Application of Unsupervised Chemometric Analysis and Self-organizing Feature Map (SOFM) for the Classification of Lighter Fuels. *Anal. Chem.*, **2010**, *82* (15), 6395–6400
13. Williams MR, Sigman ME, Lewis, J, Pitan KM. Combined target factor analysis and Bayesian soft-classification of interference-contaminated samples: Forensic Fire Debris Analysis. For. Sci. Interna. 2012, 222 (1), 373-386.
14. Waddell EE, Song ET, Rinke CN, Williams, MR, Sigman ME. Progress Toward the Determination of Correct Classification Rates in Fire Debris Analysis,, J For. Sci 2013, 58 (4), 887- 896.
15. Lu Y, Chen P, Harrington PB, Comparison of differential mobility spectrometry and mass spectrometry for gas chromatographic detection of ignitable liquids from fire debris using projected difference resolution. Anal Bioanal Chem 2009; 394:2061–2067.
16. Fire Laboratory Standards and Protocols Committee Sci., Technical Working Group for Fire and Explosives. Quality Assurance Guide for the Forensic Analysis of Ignitable Liquids. Vol. 8, Num.2, 2006.
17. ASTM 1412-12 Standard Practice for Separation of Ignitable Liquid Residues from Fire Debris Samples by Passive Headspace Concentration with Activated Charcoal, ASTM Annual Book of Standard, Vol. 14.02, ASTM International, West Conshohocken, PA, 2012.
18. Committee on Identifying the Needs of the Forensic Sciences Community, National Research Council, Strengthening Forensic Science in the United States: A Path Forward, National Academy Press, USA, 2009.
19. Hendrikse J. ENFSI collaborative testing programme for ignitable liquid analysis: A review. Forensic Sci. Internat. 2007; 167: 213–219.
20. Van De Mark TP, Bodnar MA, McGuffin VL, Smith RW. Effects of matrix interferences on the identification of mixed liquids in fire debris using Chemometrics. Proc. Am. Acad. For. Sci. 2010; Abstract A104.
21. Stauffer E, Dolan JA, Newman R. *Fire Debris Analysis*. Academic Press, USA, 2008.
22. Newman R, Dietz W, Lothridge K. The use of activated charcoal strips for fire debris extractions by passive diffusion. Part I: the effects of time, temperature, strips size, and sample concentration. J Forensic Sci 1996; 41(3):361–70.
23. Almirall JR, Furton KG, Characterization of background and pyrolysis products that may interfere with the forensic analysis of fire debris, J. Anal. Appl. Pryol. 2004; 71 (1): 51–67.
24. Lentini JJ, Dolan JA, Cherry C. The petroleum laced background. J Forensic Sci 2000; 45(5):968–989.
25. Wells SB, The identification of isopar H in vinyl flooring, J. Forensic Sci. 2005; 50 (4): 865–872.
26. Fernandes MS, Lau CM, Wong WC, The effect of volatile residues in burnt household items on the detection of fire accelerants, Sci. Justice 2002; 42 : 7–15.
27. Lentini JJ, Persistence of floor coating solvents, J. Forensic Sci. 2001; 46 (6): 1470–1473.
28. Karter M. Fire loss in the United States During 2009. National Fire Protection Association 2010: 3-4.

29. Federal Bureau of Investigation. 2009 Uniform Crime Report on Arson. September 2010. International Association of Arson Investigators Inc. A Pocket Guide to Accelerant Evidence Collection. 2001: section IV-C.
30. O'Donnell JF. The sampling of burned areas for accelerant residue analysis. The Fire and Arson Investigator 1985 June; 35(4):18-20.
31. DeHaan JD. Kirk's Fire Investigation. 7th ed. Saddle River: Pearson. 2011: 258-79, 679.
32. Icove DJ, DeHaan JD. Forensic Fire Scene Reconstruction. 2nd ed. Saddle River: Pearson. 2009: 160-3
33. Mealy CL, Benfer ME, Gottuk DT. Fire dynamics and forensic analysis of liquid fuel fires. National Criminal Justice Reference Service. Document No. 238704. May 2012: 148-151.
34. Putorti AD Jr. Flammable and combustible liquid spill/burn patterns. National Institute of Justice, NIJ-604-00, 2010: 8-9.
35. Macomber M. Migrations of ignitable liquids in pour patterns on carpet. Am. Acad. Of For Science, 63rd Annual Mtg, Chicago IL, Feb 2011; Abstract A65.
36. Evans M, Games C. Quantifying ignitable liquids in fire debris, aka how much is in my sample. Fire and Arson Investigator. April 2012: 48-53.
37. Locke AK, Basara GJ, Sandercock PML. Evaluation of internal standards for the analysis of ignitable liquids in fire debris. J For Sci 2009; 54(2): 320-7.
38. Salgueiro PAS, Borges CMF, Bettencourt da Silva RJN. Valid internal standard technique for arson detection based on gas chromatography-mass spectrometry. Journal of Chromatography A 2012; 1257: 189-94.
39. Stauffer E. Concept of pyrolysis for fire debris analysts. Science & Justice 2003; 43(1): 29-40.
40. Lentini JJ. Scientific protocols for fire investigation. Boca Raton: CRC Press. 2006: 150.
41. R Core Team 2012, "R: A Language and Environment for Statistical Computing", Ver. 2.15.2; R Foundation for Statistical Computing, Vienna, Austria.
42. Hojsgaard 2012a, "gRim: Graphical Interaction Models", Ver. 0.1-15; S. Hojsgaard, 2012.
43. Lawrence 2010, "RGtk2: A Graphical User Interface Toolkit for R", M. Lawrence and D. Temple Lang, J. Stat. Soft., 37(8), 1-52, 2010.
44. Verzani 2012, "gWidgets: gWidgets API for building toolkit-independent, interactive GUIs", Ver. 0.0-52; J. Verzani, 2012
45. Hojsgaard 2012b, "Graphical Independence Networks with the gRain Package for R", S. Hojsgaard, J. Stat. Soft., 46(10), 1-26, 2012.
46. Theodoridis S, Koutroumbas K. Pattern Recognition. 3 ed. San Diego: Academic Press, 2006.

DissSEMINATION OF RESEARCH FINDINGS

JG Rankin, A Heeren, “Effects of Competitive Adsorption on Interpretation of Ignitable Liquid Residues in Fire Debris Analysis” MAAFS 2011, Virginia Beach, VA, May 2011, Oral Presentation.

B Sullivan; JG Rankin, DL Greely, Ignitable Liquid Residue Distribution in Pour Patterns as Affected by Substrate Type and Ignitable Liquid Class. AAFS 2012, Atlanta, GA, Feb 2012, Poster Presentation. A117.

A Heeren, JG Rankin, J Vititoe, Effects of Various Substrate Types on E1618 Pattern Classification of Ignitable Liquids Present in Fire Debris. AAFS 2012, Atlanta, GA, Feb 2012, Poster Presentation. A118.

JG Rankin, DL Greely, B Sullivan, “Ignitable liquid residue distribution in pour patterns”. Proceedings of ISFI2012, Baltimore, MD October 2012. Oral Presentation and Publication of full paper in Proceedings.

JG Rankin, DL Greely, B Sullivan, “Ignitable liquid residue distribution in pour patterns”.2nd Intern. Conf. Forensic Research and Technology, Las Vegas, NV, October 2013.

JG Rankin, K Timmons; D Eckre, KL. Rusbarsky, N D. Petraco, Development of an Expert System for Classifying Ignitable Liquid Residues in Fire Debris by ASTM E1618, 2nd Intern. Conf. Forensic Research and Technology, Las Vegas, NV, October 2013. Oral Presentation.

JG Rankin, K Timmons; D Eckre, KL Rusbarsky, N D Petraco, Development of an Expert System for Classifying Ignitable Liquid Residues in Fire Debris by ASTM E1618, AAFS Seattle, WV, February 2014, A104 Oral Presentation, Abstract 104.

APPENDIX 1: R-scripts used for analysis and GUI construction.

process.raw.data.sheets.R: Functions to read in and reformat the data output from the Agilent GS-MS instrument.

```
#----------------------------------------------------------------
#Process a spread sheet
#New for Kate Timmons new data sheets
#Also modified to reflect the shift wrt internal standard peak RT
#---------------------------------------------------------------
process.raw<-function(fpath) {

  #Turn on warning immediately go we can see which file triggers them:
  options(warn=1)

  #Grab the whole spreadsheet
  raw.dat<-read.csv(fpath,header=F)
  #print(class(raw.dat))

  #Parse spread sheet into the separate plot data
  plot1.row.idx<-which(raw.dat[,1]=="Plot 1")
  plot2.row.idx<-which(raw.dat[,1]=="Plot 2")
  plot3.row.idx<-which(raw.dat[,1]=="Plot 3")
  plot4.row.idx<-which(raw.dat[,1]=="Plot 4")
  plot5.row.idx<-which(raw.dat[,1]=="Plot 5")
  plot6.row.idx<-which(raw.dat[,1]=="Plot 6")

  plot1.dat<-raw.dat[(plot1.row.idx+3):(plot2.row.idx-1),]
  plot2.dat<-raw.dat[(plot2.row.idx+3):(plot3.row.idx-1),]
  plot3.dat<-raw.dat[(plot3.row.idx+3):(plot4.row.idx-1),]
  plot4.dat<-raw.dat[(plot4.row.idx+3):(plot5.row.idx-1),]
  plot5.dat<-raw.dat[(plot5.row.idx+3):(plot6.row.idx-1),]
  plot6.dat<-raw.dat[(plot6.row.idx+3):(nrow(raw.dat)),]

  colnames(plot1.dat)<-c("Peak#", "Ret Time (min.)", "Area", "%Tot", "Sig/Noise", "Scan Descrip")
  colnames(plot2.dat)<-c("Peak#", "Ret Time (min.)", "Area", "%Tot", "Sig/Noise", "Scan Descrip")
  colnames(plot3.dat)<-c("Peak#", "Ret Time (min.)", "Area", "%Tot", "Sig/Noise", "Scan Descrip")
  colnames(plot4.dat)<-c("Peak#", "Ret Time (min.)", "Area", "%Tot", "Sig/Noise", "Scan Descrip")
  colnames(plot5.dat)<-c("Peak#", "Ret Time (min.)", "Area", "%Tot", "Sig/Noise", "Scan Descrip")
  colnames(plot6.dat)<-c("Peak#", "Ret Time (min.)", "Area", "%Tot", "Sig/Noise", "Scan Descrip")

  #Internal standard info now listed on top of each file. The I.S. couts are element 3 in
raw.dat:
  #The I.S. retention time is element 2 in raw.dat:
  #Normalize all peaks to internal standard counts:
  intern.std.counts<-as.numeric(as.character(raw.dat[1,3]))
  if(abs(intern.std.counts)==Inf) {
    print(paste("**************Warning: NO REFERENCE PEAK FOUND****************  ",fpath))
    print("---------------------------------------------------------------------------------
-")
  }
  intern.std.RT<-as.numeric(as.character(raw.dat[1,2]))

  #For each plot, just grab the retion times, corredsponding peak areas and peak % in the plot
  #**After the peak areas I am inserting R.T.s shifted wrt the I.S. peak
  shifted.rt.plot1 <- as.numeric(as.character(plot1.dat[,2])) - intern.std.RT
  shifted.rt.plot2 <- as.numeric(as.character(plot2.dat[,2])) - intern.std.RT
  shifted.rt.plot3 <- as.numeric(as.character(plot3.dat[,2])) - intern.std.RT
  shifted.rt.plot4 <- as.numeric(as.character(plot4.dat[,2])) - intern.std.RT
  shifted.rt.plot5 <- as.numeric(as.character(plot5.dat[,2])) - intern.std.RT
  shifted.rt.plot6 <- as.numeric(as.character(plot6.dat[,2])) - intern.std.RT
  #Format:   R.T.s,                                    Peak Area,                            R.T.s
shifted wrt I.S., %Total
```

```
  gc1<-cbind(as.numeric(as.character(plot1.dat[,2])), as.numeric(as.character(plot1.dat[,3])),
shifted.rt.plot1 ,as.numeric(as.character(plot1.dat[,4])))
  gc2<-cbind(as.numeric(as.character(plot2.dat[,2])), as.numeric(as.character(plot2.dat[,3])),
shifted.rt.plot2 ,as.numeric(as.character(plot2.dat[,4])))
  gc3<-cbind(as.numeric(as.character(plot3.dat[,2])), as.numeric(as.character(plot3.dat[,3])),
shifted.rt.plot3 ,as.numeric(as.character(plot3.dat[,4])))
  gc4<-cbind(as.numeric(as.character(plot4.dat[,2])), as.numeric(as.character(plot4.dat[,3])),
shifted.rt.plot4 ,as.numeric(as.character(plot4.dat[,4])))
  gc5<-cbind(as.numeric(as.character(plot5.dat[,2])), as.numeric(as.character(plot5.dat[,3])),
shifted.rt.plot5 ,as.numeric(as.character(plot5.dat[,4])))
  gc6<-cbind(as.numeric(as.character(plot6.dat[,2])), as.numeric(as.character(plot6.dat[,3])),
shifted.rt.plot6 ,as.numeric(as.character(plot6.dat[,4])))

  gc1[,2]<-gc1[,2]/intern.std.counts
  gc2[,2]<-gc2[,2]/intern.std.counts
  gc3[,2]<-gc3[,2]/intern.std.counts
  gc4[,2]<-gc4[,2]/intern.std.counts
  gc5[,2]<-gc5[,2]/intern.std.counts
  gc6[,2]<-gc6[,2]/intern.std.counts

  min.rt<-min(c(gc1[,1],gc2[,1],gc3[,1],gc4[,1],gc5[,1],gc6[,1]),na.rm = TRUE)
  max.rt<-max(c(gc1[,1],gc2[,1],gc3[,1],gc4[,1],gc5[,1],gc6[,1]),na.rm = TRUE)

  min.shifted.rt<-min(c(gc1[,3],gc2[,3],gc3[,3],gc4[,3],gc5[,3],gc6[,3]),na.rm = TRUE)
  max.shifted.rt<-max(c(gc1[,3],gc2[,3],gc3[,3],gc4[,3],gc5[,3],gc6[,3]),na.rm = TRUE)

  info<-list(gc1,gc2,gc3,gc4,gc5,gc6, c(min.rt,max.rt), c(min.shifted.rt,max.shifted.rt),
intern.std.RT)
  names(info)<-c("T.I.C.",
                "Alkanes: Exctacted ions-57-71-85-99",
                "Cycloalkanes: Exctacted ions-55-69-82-83",
                "Aromatics: Exctacted ions-91-92-105-106-119-120",
                "Alkylnapthalenes: Exctacted ions-128-142-156-170",
                "Indanes: Exctacted ions-117-118-131-132",
                "R.T.-min/max",
                "Shifted R.T.-min/max",
                "Internal Standard R.T.")

  #Put the warnings back to default
  options(warn=0)

  return(info)

}


#---------------------------------------------
#Plot chromatograms comprising a spread sheet
#---------------------------------------------
plot.gc.info<-function(gc.info,gc.nme,gc.cls) {

  gc1<-gc.info[[1]]
  gc2<-gc.info[[2]]
  gc3<-gc.info[[3]]
  gc4<-gc.info[[4]]
  gc5<-gc.info[[5]]
  gc6<-gc.info[[6]]

  par(mfrow=c(3,2))
  plot(gc1[,1],gc1[,2],typ="h", ylab="Intes.",xlab="RT (min)", main=paste(gc.nme,":
",names(gc.info)[1],", E1618 class: ",gc.cls,sep=""))

  if(NA %in% gc2[1,]) {
    print(paste("NO PLOT 2: ", gc.nme,": ",names(gc.info)[2],sep=""))
  } else {
    #print(names(gc.info)[2])
    plot(gc2[,1],gc2[,2],typ="h", ylab="Intes.",xlab="RT (min)", main=paste(gc.nme,":
",names(gc.info)[2],sep=""))
  }
```

```
  if(NA %in% gc3[1,]) {
    print(paste("NO PLOT 3: ", gc.nme,": ",names(gc.info)[3],sep=""))
  } else {
    plot(gc3[,1],gc3[,2],typ="h", ylab="Intes.",xlab="RT (min)", main=paste(gc.nme,":
",names(gc.info)[3],sep=""))
  }

  if(NA %in% gc4[1,]) {
    print(paste("NO PLOT 4: ", gc.nme,": ",names(gc.info)[4],sep=""))
  } else {
    plot(gc4[,1],gc4[,2],typ="h", ylab="Intes.",xlab="RT (min)", main=paste(gc.nme,":
",names(gc.info)[4],sep=""))
  }

  if(NA %in% gc5[1,]) {
    print(paste("NO PLOT 5: ", gc.nme,": ",names(gc.info)[5],sep=""))
  } else {
    plot(gc5[,1],gc5[,2],typ="h", ylab="Intes.",xlab="RT (min)", main=paste(gc.nme,":
",names(gc.info)[5],sep=""))
  }

  if(NA %in% gc6[1,]) {
    print(paste("NO PLOT 6: ", gc.nme,": ",names(gc.info)[6],sep=""))
  } else {
    plot(gc6[,1],gc6[,2],typ="h", ylab="Intes.",xlab="RT (min)", main=paste(gc.nme,":
",names(gc.info)[6],sep=""))
  }

}


#----------------------------------------------
#Plot chromatograms comprising a spread sheet
#----------------------------------------------
plot.gc.info2<-function(gc.info,gc.nme,gc.cls,axis.typ="shifted") {

  gc1<-gc.info[[1]]
  gc2<-gc.info[[2]]
  gc3<-gc.info[[3]]
  gc4<-gc.info[[4]]
  gc5<-gc.info[[5]]
  gc6<-gc.info[[6]]

  par(mfrow=c(3,2))
  if(axis.typ=="shifted"){
    plot(gc1[,3],gc1[,2],typ="h", ylab="Intes.",xlab="Shifted RT wrt I.S. (min)",
main=paste(gc.nme,": ",names(gc.info)[1],", E1618 class: ",gc.cls,sep=""))
  } else {
    plot(gc1[,1],gc1[,2],typ="h", ylab="Intes.",xlab="RT (min)", main=paste(gc.nme,":
",names(gc.info)[1],", E1618 class: ",gc.cls,sep=""))
  }

  if(NA %in% gc2[1,]) {
    print(paste("NO PLOT 2: ", gc.nme,": ",names(gc.info)[2],sep=""))
  } else {
    #print(names(gc.info)[2])
    if(axis.typ=="shifted"){
      plot(gc2[,3],gc2[,2],typ="h", ylab="Intes.",xlab="Shifted RT wrt I.S. (min)",
main=paste(gc.nme,": ",names(gc.info)[2],", E1618 class: ",gc.cls,sep=""))
    } else {
      plot(gc2[,1],gc2[,2],typ="h", ylab="Intes.",xlab="RT (min)", main=paste(gc.nme,":
",names(gc.info)[2],", E1618 class: ",gc.cls,sep=""))
    }
  }

  if(NA %in% gc3[1,]) {
    print(paste("NO PLOT 3: ", gc.nme,": ",names(gc.info)[3],sep=""))
  } else {
```

```
    if(axis.typ=="shifted"){
      plot(gc3[,3],gc3[,2],typ="h", ylab="Intes.",xlab="Shifted RT wrt I.S. (min)",
main=paste(gc.nme,": ",names(gc.info)[3],", E1618 class: ",gc.cls,sep=""))
    } else {
      plot(gc3[,1],gc3[,2],typ="h", ylab="Intes.",xlab="RT (min)", main=paste(gc.nme,":
",names(gc.info)[3],", E1618 class: ",gc.cls,sep=""))
    }
  }

  if(NA %in% gc4[1,]) {
    print(paste("NO PLOT 4: ", gc.nme,": ",names(gc.info)[4],sep=""))
  } else {
    if(axis.typ=="shifted"){
      plot(gc4[,3],gc4[,2],typ="h", ylab="Intes.",xlab="Shifted RT wrt I.S. (min)",
main=paste(gc.nme,": ",names(gc.info)[4],", E1618 class: ",gc.cls,sep=""))
    } else {
      plot(gc4[,1],gc4[,2],typ="h", ylab="Intes.",xlab="RT (min)", main=paste(gc.nme,":
",names(gc.info)[4],", E1618 class: ",gc.cls,sep=""))
    }
  }

  if(NA %in% gc5[1,]) {
    print(paste("NO PLOT 5: ", gc.nme,": ",names(gc.info)[5],sep=""))
  } else {
    if(axis.typ=="shifted"){
      plot(gc5[,3],gc5[,2],typ="h", ylab="Intes.",xlab="Shifted RT wrt I.S. (min)",
main=paste(gc.nme,": ",names(gc.info)[5],", E1618 class: ",gc.cls,sep=""))
    } else {
      plot(gc5[,1],gc5[,2],typ="h", ylab="Intes.",xlab="RT (min)", main=paste(gc.nme,":
",names(gc.info)[5],", E1618 class: ",gc.cls,sep=""))
    }
  }

  if(NA %in% gc6[1,]) {
    print(paste("NO PLOT 6: ", gc.nme,": ",names(gc.info)[6],sep=""))
  } else {
    if(axis.typ=="shifted"){
      plot(gc6[,3],gc6[,2],typ="h", ylab="Intes.",xlab="Shifted RT wrt I.S. (min)",
main=paste(gc.nme,": ",names(gc.info)[6],", E1618 class: ",gc.cls,sep=""))
    } else {
      plot(gc6[,1],gc6[,2],typ="h", ylab="Intes.",xlab="RT (min)", main=paste(gc.nme,":
",names(gc.info)[6],", E1618 class: ",gc.cls,sep=""))
    }
  }
}



#--------------------------------------
#Construct file paths to spread sheets
#--------------------------------------
build.file.paths<-function(root.dir,sub.dir,samp.names,samp.reps) {

  #Sys.info()
  if(.Platform$OS.type=="unix") {
    file.sep<-"/"
  } else {
    file.sep<-"\\"
  }

  file.paths<-NULL
  for(i in 1:length(samp.names)) {
    samp.name<-samp.names[i]
    for(j in 1:samp.reps[i]) {

      file.path<-paste(root.dir,sub.dir,file.sep,samp.name,"-",j,".csv",sep="")
      file.paths<-rbind(file.paths,file.path)
    }
  }
```

```
  return(file.paths)

}

#--------------------------------
#Process a batch of spread sheets
#--------------------------------
process.batch<-function(file.paths) {
  container<-rep(list(NULL), length(file.paths))
  for(i in 1:length(file.paths)) {
    print(file.paths[[i]])
    container[[i]] <- process.raw(file.paths[[i]])
  }

  return(container)

}


#----------------------------------------------------------
#All the same ion plots for all the replicates of a species
#----------------------------------------------------------
plot.ion.group<-function(grp.num,plot.num,samp.names,samp.num.reps,samp.container,printQ=FALSE) {

  lbls.vec<-generate.label.vec(samp.names, samp.num.reps)
  grp.start.idx<-which(lbls.vec==grp.num)[1]
  grp.end.idx<-grp.start.idx+length(which(lbls.vec==grp.num))-1

  grp.idx<-which(samp.names==grp.num)

  #Collect the sub ion-info across the replicates in a group together:
  gcpl<-rep(list(NULL),samp.num.reps[grp.idx])
  count<-1
  col1.cont<-NULL
  col2.cont<-NULL
  for(i in grp.start.idx:grp.end.idx) {
    gci<-samp.container[[i]]
    gcp<-gci[[plot.num]]
    gcpl[[count]]<-gcp
    count<-count+1
  }
  sub.plot.ions<-names(gci)[plot.num] #Just take from last container examined. All should be the
same.

  #Sub-plots R.T. min/max
  options(warn=-1) #Shut off annoying warning for Inf substitutions in empty vectors. These NA
vects won't  get printed anyway.
  plot.rt.min<-min(unlist(sapply(1:length(gcpl),function(x){gcpl[[x]][,1]})),na.rm=TRUE)
  plot.rt.max<-max(unlist(sapply(1:length(gcpl),function(x){gcpl[[x]][,1]})),na.rm=TRUE)
  #Sub-plots intensitiy min/max
  plot.inten.min<-min(unlist(sapply(1:length(gcpl),function(x){gcpl[[x]][,2]})),na.rm=TRUE)
  plot.inten.max<-max(unlist(sapply(1:length(gcpl),function(x){gcpl[[x]][,2]})),na.rm=TRUE)
  options(warn=0)
  #print(c(plot.rt.min,plot.rt.max,plot.inten.min,plot.inten.max))

  print(paste(length(gcpl), "replicates for", grp.num, sub.plot.ions))
  plot.count<-0
  par(mfrow=c(3,2))
  for(i in 1:length(gcpl)) {
    gcp<-gcpl[[i]]
    #print(gcp)

    if(NA %in% gcp[1,]) {
      print(paste("NO PLOT FOR: ",grp.num,"-",i,": ",sub.plot.ions,sep=""))
    } else {
      if(sum(gcp[,2]==0) == length(gcp[,2])) {
```

```
        plot(gcp[,1],gcp[,2],typ="h", main=paste(grp.num,"-",i,": ",sub.plot.ions," WARNING ALL
Intes. 0!",sep=""), ylab="Intes.", xlab="RT (min)", xlim=c(plot.rt.min,plot.rt.max),
ylim=c(plot.inten.min,plot.inten.max))
      } else {
        plot(gcp[,1],gcp[,2],typ="h", main=paste(grp.num,"-",i,": ",sub.plot.ions,sep=""),
ylab="Intes.", xlab="RT (min)", xlim=c(plot.rt.min,plot.rt.max),
ylim=c(plot.inten.min,plot.inten.max))
      }

      #print(gcp)
      plot.count<-plot.count+1
    }
  }

  if(printQ==TRUE) {
    print(gcpl)
  }

}

#--------------------------------------------------------------
#Plot all the chromatograms in series for error checking.
#--------------------------------------------------------------
plot.all.gc<-function(info.container,names.mat, starting.idx=1) {

  if(names(dev.cur())=="null device") {
    #print("IM HERE!")
    dev.new()
  }

  for(i in starting.idx:length(info.container)) {
    inp<-readline(prompt="Hit enter for next plot set, q to quit... ")
    if(inp=="q") {
      print("Quitting Loop")
      break()
    }
    dev.off()
    gc.name<-as.character(names.mat[i,2])
    gc.name.orig<-as.character(names.mat[i,1])
    gc.class<-as.character(names.mat[i,4])
    gc.info<-info.container[[i]]
    print(paste(i,":",gc.name.orig,"    Re-named:",gc.name,", E1618 Class:",gc.class))
    plot.gc.info(gc.info, gc.name, gc.class)
  }

}


#--------------------------------------------------------------
#Plot all the EIC plots in a Chromatogram.
#--------------------------------------------------------------
plot.all.ion.plots<-function(grp.idx, name.vec, reps.vec, info.container, printQ=FALSE) {

  for(plot.num in 1:6) {
    inp<-readline(prompt="Hit enter for next ion set, q to quit... ")
    if(inp=="q") {
      print("Quitting Loop")
      break()
    }

    if(!(names(dev.cur())=="null device")) {
      dev.off()
    }

    print(paste("Plot #:",plot.num))
    #These should all look similar. Do they??
    plot.ion.group(grp.idx, plot.num, samp.names=name.vec, samp.num.reps=reps.vec,
samp.container=info.container,printQ)
  }
```

```
}


#-------------------------------------------------------------------
#Counts the number of replicates for each group.
#Should be independent of group naming convention
#-------------------------------------------------------------------
count.group.replicates<-function(arb.lbls) {
  char.lbls<-as.character(arb.lbls)
  lbl.names<-unique(char.lbls)
  num.samps.vec<-sapply(1:length(lbl.names),function(x){sum(char.lbls==lbl.names[x])})
  #num.samps.vec<-sapply(as.numeric(levels(factor(arb.lbls))), function(x){sum(arb.lbls==x)})
  return(num.samps.vec)

}

#-------------------------------------------------------------------
#Pick out groups of observations and form a new X matrix
#-------------------------------------------------------------------
pick.out.groups<-function(X.mat, all.arb.lbls, grp.picks)
{

  pick.out.rows<-NULL
  new.grp.lbls<-NULL
  char.lbls<-as.character(all.arb.lbls)
  char.grp.picks<-as.character(grp.picks)

  for(i in 1:length(grp.picks))
  {
    grp.idxs<-which(char.lbls==char.grp.picks[i])
    pick.out.rows<-c(pick.out.rows,grp.idxs)
    print(grp.idxs)
    new.grp.lbl<-rep(i,length(grp.idxs))
    new.grp.lbls<-c(new.grp.lbls,new.grp.lbl)
  }

  new.grp.lbls<-factor(new.grp.lbls)
  new.X.mat<-X.mat[pick.out.rows,]

  return(list(new.X.mat,new.grp.lbls))

}


#----------------------------------------------------
#Parse out useful info from the loaded name.csv file
#----------------------------------------------------
parse.name.file<-function(original.names.mat) {

  item.names<-
t(sapply(1:nrow(original.names.mat),function(x){strsplit(as.character(original.names.mat[x,2]),
split="-")[[1]]}))[,1]
  species.names<-unique(item.names)
  species.reps<-count.group.replicates(item.names)

  names.info<-list(item.names, species.names, species.reps)

  names(names.info)<-c("Obs. Names", "Species Names", "Num. Reps.")

  return(names.info)

}


#-------------------------------------------------------------------
#Make up a lable vector according to number of samples
#-------------------------------------------------------------------
generate.label.vec<-function(samp.names.vec, num.samps.vec)
```

```
{

  lbl.vec<-NULL
  for(i in 1:length(num.samps.vec) )
  {
    grpids<-rep(samp.names.vec[i],num.samps.vec[i])
    lbl.vec<-c(lbl.vec,grpids)
  }

  return(lbl.vec)

}


#--------------------------------------------------------------
#Bin the time axes. Too slow! Not used currently.
#--------------------------------------------------------------
bin.time.axes<-function(name.vec, reps.vec, info.container, delt.frac=0.1, printQ=FALSE) {

  plot.info.vecs<-rep(list(NULL),6) #times listed for plot
  dts<-rep(100000000000000,6)
  for(i in 1:length(info.container)) {
    sub.info<-info.container[[i]]
    for(j in 1:6) {
      plot.times<-sub.info[[j]][,1] #Grab the time values
      #dts[j]<-  min(dts[j], min(diff(plot.times)))
      #print(min(diff(plot.times)))
      options(warn=-1)
      dts[j] <- min(dts[j], min(diff(plot.times),na.rm=T))
      options(warn=0)
      plot.info.vecs[[j]]<-c(plot.info.vecs[[j]],plot.times)
    }
  }
  #print(dts)


  #print(plot.info.vecs)
  options(warn=-1)
  t.maxs<-sapply(1:6,function(x){max(plot.info.vecs[[x]], na.rm=TRUE)})
  t.mins<-sapply(1:6,function(x){min(plot.info.vecs[[x]], na.rm=TRUE)})
  options(warn=0)
  time.params<-cbind(1:6,t.mins,t.maxs,dts)
  colnames(time.params)<-c("plot#","Time min","Time max", "delta-time")
  #print(time.params)
  if(Inf %in% abs(t.maxs)) {
    empty.idxs<-which(abs(t.maxs)==Inf)
    print("Dropping plots#:")
    print(empty.idxs)
    print("No data!")
    time.params <- time.params[-empty.idxs,]
    #dts <- dts[-empty.idxs]
  }
  print("Time axis parameters:")
  print(time.params)

  time.bins<-rep(list(NULL),nrow(time.params))
  names(time.bins)<-c(paste("Plot#:", time.params[,1], "time axis"))
  for(i in 1:nrow(time.params)) {
    ax<-seq(from=floor(time.params[i,2]), to=ceiling(time.params[i,3]), by=(time.params[i,4]-
delt.frac*time.params[i,4]) )
    time.bins[[i]]<-ax
  }
  #print(time.bins)
  names(time.bins)<-c(time.params[,1]) #These are INDEXES of the time axes, not the plot numbers.
  #print(names(time.bins))

  all.fvs<-NULL
  plot1.fvs<-NULL
  for(i in 1:length(info.container)) {
```

```
    specn.info<-info.container[[i]]

    total.fv<-NULL
    #loop over all the plots with atleast one rt in the info.container:
    for(j in time.params[,1]) {
      plot.info<-specn.info[[j]]

      obs.rts<-plot.info[,1] #Retention times of plot

      obs.normed.ints<-plot.info[,2] #Intensities corresponding to retention times of plot

      time.ax.idx<-which(names(time.bins)==j) #Get a generic time axis for the particular plot
      t.axis<-time.bins[[time.ax.idx]]
      #print(t.axis)

      #Make fv for plot
      if(NA %in% obs.rts) { #make an empty fv if NAs in plot (all elements should be NAs)
        fv<-numeric(length(t.axis))
        total.fv<-c(total.fv,fv)
      } else { #else make an fv with intensities in the corresponding time bins and zeros ew
        fv<-numeric(length(t.axis))
        for(k in 1:length(obs.rts)) {
          it.idx<-(which((obs.rts[k]<=t.axis)==TRUE)[1])

          #Some problems encountered can be be indicated here:
          if(is.na(it.idx)) {
            print("NA bin indices encountered!!!!!:")
            print(paste("Container",i,"Plot",j))
            print(obs.rts[k])
            print(max(t.axis))
          }
          #put this in an else??
          fv[it.idx]<-obs.normed.ints[k]
        }
        if(j!=1) {
          total.fv<-c(total.fv,fv) #stack selective ion chromatograms
        } else {
          if(j==1) {
            plot1.fv<-fv #total ion chromatogram
          }
        }
      }
    }
    #print(length(total.fv))
    all.fvs<-rbind(all.fvs,total.fv)
    plot1.fvs<-rbind(plot1.fvs, plot1.fv)
  }

  fv.info<-list(all.fvs,plot1.fvs,time.bins)
  names(fv.info)<-c("Stacked Selective Ion FVs","TIC FVs", "Time Axes")
  return(fv.info)

}


#--------------------------------------------------------------
#Older Plot FVs
#NOTE: arb.lbls MUST take any reduction in dmat into account!
#--------------------------------------------------------------
plot.group.fvs<-function(dmat,arb.lbls,group.pick,rt.axes) {

  dmatg<-pick.out.groups(dmat,arb.lbls,c(group.pick))[[1]]
  obs.idxs<-which(arb.lbls==group.pick)

  if(!(names(dev.cur())=="null device")) {
    dev.off()
  }

  #Find breaks in ploted axes
```

```
  breks<-NULL
  st<-1
  for(i in 1:length(rt.axes)) {
    if(names(rt.axes)[i]!="1") {
      sp<-length(rt.axes[[i]])+st-1
      breks<-rbind(breks, as.numeric(c(names(rt.axes)[i],st,sp)))
      st<-sp+1
    }
  }
  #print(breks)

  par(mfrow=c(3,2)) #Caution: Assumes at most 6 replicates
  for(i in 1:nrow(dmatg)) {
   titl<-paste("Spec#:",group.pick,"Curr Rep#:",i,"Curr Obs#:",obs.idxs[i])
   plot(dmatg[i,],typ="h",main=titl)
   for(j in 1:nrow(breks)){
     abline(v = breks[j,3], col = "blue",lty=2)
   }
  }

}

#-----------------------------------------------------------------
#Plot FVs
#NOTE: arb.lbls MUST take any reduction in dmat into account!
#-----------------------------------------------------------------
plot.group.fvs2<-function(dmat,arb.lbls,class.lbls,group.pick,rt.axes) {

  dmatg<-pick.out.groups(dmat,arb.lbls,c(group.pick))[[1]]
  obs.idxs<-which(arb.lbls==group.pick)

  class.lbl<-class.lbls[obs.idxs][1]

  if(!(names(dev.cur())=="null device")) {
    dev.off()
  }

  #Find breaks in ploted axes
  breks<-NULL
  st<-1
  for(i in 1:length(rt.axes)) {
    if(names(rt.axes)[i]!="1") {
      sp<-length(rt.axes[[i]])+st-1
      breks<-rbind(breks, as.numeric(c(names(rt.axes)[i],st,sp)))
      st<-sp+1
    }
  }
  #print(breks)
  plot.names<-c("all ions","ions-57-71-85-99","ions-55-69-82-83","ions-91-92-105-106-119-
120","ions-128-142-156-170","ions-117-118-131-132")
  breks.info<-NULL
  for(i in 1:nrow(breks)){
    breks.info<-c(breks.info,plot.names[breks[i,1]])
  }
  breks.info<-data.frame(breks[,1],breks.info,breks[,2:3])
  colnames(breks.info)<-c("Selective Ion Plot#","Ions","Idx Start","Idx Stop")
  print(breks.info)

  par(mfrow=c(3,2)) #Caution: Assumes at most 6 replicates
  for(i in 1:nrow(dmatg)) {
    titl<-paste("Spec#:",group.pick,"Curr Rep#:",i,"Curr Obs#:",obs.idxs[i])
    plot(dmatg[i,],typ="h",main=titl,xlab=paste("Class:",class.lbl))
    for(j in 1:nrow(breks)){
      abline(v = breks[j,3], col = "blue",lty=2)
    }
  }

}
```

```
#---------------------------
#Normalize a chromatogram
#---------------------------
norm.profile<-function(profile)
{

  numNAs<-length(profile)-length(na.omit(profile))

  nprofl<-na.omit(profile)
  minp<-min(nprofl)
  maxp<-max(nprofl)

  nprofl<-apply(as.array(nprofl),1,function(x){(x-minp)/(maxp-minp)})

  nprofl<-c(nprofl,rep(NA,numNAs))
  return(nprofl)

}
```

process.raw.data.sheets2.R: Functions to read in and reformat the data output from the Agilent GS-MS instrument.

```
#-----------------------------------------------------------------
#Smoothed out file path construction routine
#NOTE: requires a file with the paths in it,
#with the file seperators being commas. See
#file-names-and-control.csv file for an example
#
#file.info.mat is the file info from file-names-and-control.csv
#----------------------------------------------------------------
build.paths.to.chromatogram.files<-function(file.info.mat, root.dir){

  #Get the correct file seperator format, depending on your OS:
  if(.Platform$OS.type=="unix") {
    file.sep<-"/"
  } else {
    file.sep<-"\\"
  }

  file.paths<-rep("X",nrow(file.info.mat))
  for(i in 1:nrow(file.info.mat)){
    file.path<-paste(root.dir, file.info.mat[i,1], file.sep, file.info.mat[i,2], " ",
file.info.mat[i,3], ".csv",sep="")
    file.paths[i]<-file.path
  }

  return(file.paths)

}

#---------------------------------------------------------------
#Wrapper function to remove unwanted cratograms, which typically
#will have some kind of flaw. Chromatograms with an x in column
#4 of file.info.mat will be tossed (removed)
#---------------------------------------------------------------
remove.unwanted.chromatograms<-function(file.info.mat, current.file.paths){
  #Find which chromatograms to remove:
  toss.idxs<-which(file.info.mat[,4]=="x")
  #print(toss.idxs)

  if(length(toss.idxs)==0) {
    #Remove nothing if there were no x-es:
    pruned.file.paths<-current.file.paths
  }
  if((length(toss.idxs)>0)){
    #Remove the unwanted chromatograms from the vector of file paths:
    pruned.file.paths<-current.file.paths[-toss.idxs]
```

```
  }

  return(pruned.file.paths)

}

#-------------------------------------------------------------
#Wrapper function to do the required labeling of each input
#chromatogram with their designated ILRC class, e.g. MPD, Aromatic, etc
#-------------------------------------------------------------
make.chromatogram.ILRC.labels<-function(ilrc.classification.info, file.info.mat){

  #Just in case it got read in as a factor:
  unique.sample.numbers<-as.numeric(as.character(ilrc.classification.info[,1]))

  #Don't want to assume 5 replicates per sample. Let's count the reps per sample number:
  #Also, just in case something got read in as a factor:
  sample.number.vec<-as.numeric(as.character(file.info.mat[,2]))
  toss.idxs<-which(file.info.mat[,4]=="x")
  #print(toss.idxs)
  if(length(toss.idxs)==0){
    sample.number.vec<-sample.number.vec
  }
  if(length(toss.idxs)>0){
    sample.number.vec<-sample.number.vec[-toss.idxs]
  }

  #Count the replicates in each sample group:
  num.reps.vec<-NULL
  for(i in 1:length(unique.sample.numbers)){
    count<-0
    count<-length(which(sample.number.vec==unique.sample.numbers[i]))
    #print(which(sample.number.vec==unique.sample.numbers[i]))
    #print(paste("Number of",unique.sample.numbers[i],"=",count))
    #print("")
    num.reps.vec<-c(num.reps.vec, count)
  }

  #Proper ILRC class names corresponding to each number:
  sample.classes<-ilrc.classification.info[,4]

  #Make a label for each chromatogram. CAUTION keeping these FACTORS!
  chromatogram.lbl<-NULL
  sample.num.for.lbl<-NULL
  for(i in 1:length(sample.classes)){
    chromatogram.lbl<-c(chromatogram.lbl, rep(as.character(sample.classes[i]), num.reps.vec[i]))
    sample.num.for.lbl<-c(sample.num.for.lbl, rep(unique.sample.numbers[i], num.reps.vec[i]))
  }
  chromatogram.lbl<-as.factor(chromatogram.lbl)
  #rep.vec<-rep(1:5,length(sample.numbers))
  label.info.mat<-data.frame(sample.num.for.lbl,chromatogram.lbl)
  colnames(label.info.mat)<-c("Sample#:","Actual ILRC Class")

  num.reps.vec<-t(as.matrix(num.reps.vec))
  colnames(num.reps.vec)<-unique.sample.numbers

  label.info<-list(num.reps.vec,label.info.mat)
  names(label.info)<-c("Number Of Replicates For Each Sample Number","ILRC Class Key Table")

  return(label.info)

}

#-----------------------------------------------------------------
#Wrapper function to pick out the E1618 class info and product
#descriptions from the ILRC Reference List.csv file. Obviates the
#need to create a new ILRC Class Info.csv file which only contains
#the sample numbers to be analysed. This way all the user needs
#to do is "x-out" sample numbers in the file-names-and-control.csv
```

```
#file that they DON'T want to analyse for whatever reason.
#-----------------------------------------------------------------
get.e1618.class.info<-function(fpath.to.e1618.ref.list, file.info.mat){

  #Pick out the indices in file.info.mat for the sample numbers we wat to keep:
  idxs.of.samp.nums.to.examine<-which(file.info.mat[,4] != "x")
  #print(idxs.of.samp.nums.to.examine)

  #Now get the unique sample number identifiers:
  samp.nums.to.examine<-as.character(unique(file.info.mat[idxs.of.samp.nums.to.examine,2]))
  #print(samp.nums.to.examine)

  #print(fpath.to.e1618.ref.list)
  all.class.info<-read.csv(fpath.to.e1618.ref.list,header=T)
  #print(all.class.info)
  db.idxs<-
unlist(sapply(1:length(samp.nums.to.examine),function(x){which(all.class.info[,1]==samp.nums.to.e
xamine[x])}))
  #print(unlist(db.idxs))
  #print(all.class.info[,1])
  #all.sample.numbers<-all.class.info[,1]
  #db.idxs.of.samp.nums.to.examine<-NULL
  #for(i in 1:length(samp.nums.to.examine)){
  #  db.idxs.of.samp.nums.to.examine<-
c(db.idxs.of.samp.nums.to.examine,which(all.sample.numbers==samp.nums.to.examine[i]))
  #}

  return(all.class.info[db.idxs,c(1:4)])

}
```

process.chromatograms.R: Functions to turn the processed chromatogam data into several types of feature vectors explored in this work.

```
library(dtw)
#These are the shifts are empirical and with respect to the internal standard.
e1618.ref.vec<-matrix(c(-10.78, -9.37, -6.54, -3.47, -0.72, 1.71, 3.89, 5.87),nrow=1 )
colnames(e1618.ref.vec)<-c("C6","C8","C10","C12","C14","C16","C18","C20")
#Expand to include odd numbered C categories. +/- 100 is used only to include extreme RTs:
C.bins<-c(-100,sort(c(e1618.ref.vec,e1618.ref.vec[1:7]+(e1618.ref.vec[2:8]-
e1618.ref.vec[1:7])/2)), 100)
C.intervals<-c("(-100,-10.8]","(-10.8,-10.1]","(-10.1,-9.37]","(-9.37,-7.96]","(-7.96,-6.54]","(-
6.54,-5]","(-5,-3.47]","(-3.47,-2.1]","(-2.1,-0.72]","(-
0.72,0.495]","(0.495,1.71]","(1.71,2.8]","(2.8,3.89]","(3.89,4.88]","(4.88,5.87]","(5.87,100]")
C.levels<-
c("C6","C7","C8","C9","C10","C11","C12","C13","C14","C15","C16","C17","C18","C19","C20","C20+")


#------------------------------------------------
#Look through all the ion plots in the group.
#As a reference for each plot, choose the group
#member with the most peaks in that plot. For ties
#choose the first plot found as the reference
#------------------------------------------------
find.references<-function(dmat,rt.axs,print.lvl=0) {

  #SPLT of FV into SEPARATE ION DATA BEFORE ALIGNMENT
  #Find breaks in ploted axes
  breks<-NULL
  st<-1
  for(i in 1:length(rt.axs)) {
    if(names(rt.axs)[i]!="1") { #Don't worry about the first plot. It's the TIC
      sp<-length(rt.axs[[i]])+st-1
      plot.grp<-dmat[,st:sp]
```

```
      peak.count.vec<-apply(plot.grp,1,peak.count)
      plot.ref.idx<-which(peak.count.vec==max(peak.count.vec))[1]
      breks<-rbind(breks, as.numeric(c(names(rt.axs)[i],st,sp,plot.ref.idx,max(peak.count.vec))))

      if(print.lvl>1) {
        print("Peak counts for Plots:",names(rt.axs[i]))
        print(peak.count.vec)
        print(paste("Reference obs for plot",names(rt.axs[i]),"is:"))
        print(plot.ref.idx)
        print("xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx")
      }

      st<-sp+1

    }
  }
  colnames(breks)<-c("Plot#","Start","Stop","Ref Obs#","#Peaks in Ref")
  if(print.lvl>0) {
    print(breks)
  }

  return(breks)

}


#----------------------------------------------
#Loop over groups
#for a group, pull out ref ions plots
#tack reference ion plots together for each group
#----------------------------------------------
construct.references<-function(dmat, arb.lbls, rt.axs) {

  #print(arb.lbls)
  lbl.names<-unique(arb.lbls)

  ref.mat<-array(0,c(length(lbl.names), ncol(dmat)))
  for(i in 1:length(lbl.names)) {
    grp.pic<-lbl.names[i]
    grp.idxs<-which(arb.lbls==grp.pic)
    grp.mat<-t(apply(dmat[grp.idxs,],1,norm.profile))
    #grp.mat<-pick.out.groups(dmat,arb.lbls,c(grp.pic))[[1]]
    #print(dim(grp.mat))
    grp.ref.mat<-find.references(grp.mat,rt.axs,0)
    print(paste("Group:",grp.pic))
    #print(which(arb.lbls==grp.pic))
    print(grp.ref.mat)
    #print("==============================")

    #Loop over plots for the group:
     grp.ref.fv<-NULL
     for(j in 1:nrow(grp.ref.mat)) {
       st<-grp.ref.mat[j,2]
       #print(st)
       sp<-grp.ref.mat[j,3]
       ref.idx<-grp.ref.mat[j,4]
       grp.ref.fv<-c(grp.ref.fv, grp.mat[ref.idx,st:sp])
       #print(paste("Group:",lbl.names[i],st,sp,ref.idx,"Num
peaks:",peak.count(grp.mat[ref.idx,st:sp]) ))
       #if((lbl.names[i]=="241") & (j==2) ){
       #   plot(grp.mat[ref.idx,st:sp],typ="h",main=paste("TEST:",lbl.names[i]))
       # }
     }
    #print("==============================")
    ref.mat[i,]<-grp.ref.fv

  }

  return(ref.mat)
```

```
}


#------
#
#------
reduce<-function(qury,ref,typ="ends") {

  #Get peak indices:
  qury.peak.idxs<-which(qury>0)
  ref.peak.idxs<-which(ref>0)

  #Chop off common zeros at ends option:
  if(typ=="ends") {

    #There are peaks in both the query and reference => bother to set them up to be compares
    if((length(qury.peak.idxs)>0) & (length(ref.peak.idxs)>0) ) {

      left.idx<-min(c(qury.peak.idxs,ref.peak.idxs))
      right.idx<-max(c(qury.peak.idxs,ref.peak.idxs))

      reduced.qury<-qury[left.idx:right.idx]
      reduced.ref<-ref[left.idx:right.idx]

      return(rbind(reduced.qury,reduced.ref))

    }
    #If one or both the signals do not have peaks, send the message not to bother to compare
    #Two cases:
    #1. One has no peak's while the other does. They can not be mapped to each other. return 1e8
    if(xor((length(qury.peak.idxs)==0),(length(ref.peak.idxs)==0))) {
      return(1e8)
    }
    #2. Both have no peaks. They are the "same" but contain no info. return -1
    if((length(qury.peak.idxs)==0) & (length(ref.peak.idxs)==0)) {
      return(-1)
    }
  }

  #Remove all common zeros option:
  if(typ=="zeros") {

    #There are peaks in both the query and reference => bother to set them up to be compares
    if((length(qury.peak.idxs)>0) & (length(ref.peak.idxs)>0) ) {

      drop.idxs<-which(colSums(rbind(qury,ref))==0)
      reduced.qury<-qury[-drop.idxs]
      reduced.ref<-ref[-drop.idxs]

      return(rbind(reduced.qury,reduced.ref))
    }
    #If one or both the signals do not have peaks, send the message not to bother to compare
    #Two cases:
    #1. One has no peak's while the other does. They can not be mapped to each other. return 1e8
    if(xor((length(qury.peak.idxs)==0),(length(ref.peak.idxs)==0))) {
      return(1e8)
    }
    #2. Both have no peaks. They are the "same" but contain no info. return -1
    if((length(qury.peak.idxs)==0) & (length(ref.peak.idxs)==0)) {
      return(-1)
    }
  }

  #Remove all common zeros option:
  #Don't do anything option:
  if(typ=="none") {
    #There are peaks in both the query and reference => bother to set them up to be compares
    if((length(qury.peak.idxs)>0) & (length(ref.peak.idxs)>0) ) {
```

```
      reduced.qury<-qury
      reduced.ref<-ref

      return(rbind(reduced.qury,reduced.ref))
    }
    #If one or both the signals do not have peaks, send the message not to bother to compare
    #Two cases:
    #1. One has no peak's while the other does. They can not be mapped to each other. return 1e8
    if(xor((length(qury.peak.idxs)==0),(length(ref.peak.idxs)==0))) {
      return(1e8)
    }
    #2. Both have no peaks. They are the "same" but contain no info. return -1
    if((length(qury.peak.idxs)==0) & (length(ref.peak.idxs)==0)) {
      return(-1)
    }
  }

}

#---------------------------------------------
#Use  dtw to warp/align test to reference
#---------------------------------------------
align.peaks<-function(tes,ref,normQ=F,plotQ=F,printQ=F, ...){

  #CAUTION: Assumes tes and ref are the same length!
  dmat<-rbind(ref,tes)
  drop.idxs<-which(colSums(dmat)==0)
  ref.idxs<-1:length(ref)

  dmat<-rbind(ref.idxs,dmat)
  dmat<-dmat[,-drop.idxs]
  #print(dmat)

  tes.sig<-dmat[3,]
  ref.sig<-dmat[2,]

  if(normQ==TRUE) {
    tes.sig<-norm.profile(tes.sig)
    ref.sig<-norm.profile(ref.sig)
  }

  #DTW
  algn<-dtw(tes.sig,ref.sig,...)
  #plot(algn)

  xx<-algn$index1
  yy<-algn$index2

  warped.tes.sig<-numeric(length(ref.sig))
  for(i in 1:length(xx)) {
    warped.tes.sig[yy[i]] <- tes.sig[xx[i]]
  }
  cs<-cor(ref.sig,warped.tes.sig)
  cstt<-cor(tes.sig,warped.tes.sig)

  full.leng.warped.sig<-numeric(length(ref))
  ref.idxs<-dmat[1,]
  #print(ref.idxs)
  for(i in 1:length(ref.idxs)) {
    full.leng.warped.sig[ref.idxs[i]] <- warped.tes.sig[i]
  }
  csr<-cor(ref,full.leng.warped.sig)
  cst<-cor(tes,full.leng.warped.sig)
  #par(mfrow=c(2,1))
  #plot(tes,typ="h")
  #plot(full.leng.warped.sig,typ="h",col="blue")
  #par(mfrow=c(1,1))
```

```
  if(plotQ==TRUE) {

    for(i in 1:2) {
      inp<-readline(prompt="Hit enter for next plot set, q to quit... ")
      if(inp=="q") {
        print("Quitting Loop")
        break()
      }
      if(i==1) {
        dtwPlotTwoWay(algn,tes.sig,ref.sig,ts.typ="h",main="Ref = red, Mov = black")
      }
      if(i==2) {
        par(mfrow=c(2,1))
        plot(ref.sig,typ="h",col="green",lwd=6,main="Before (Ref = green, Mov = blue)")
        lines(tes.sig,typ="h",col="blue",lwd=2)

        plot(ref.sig,typ="h",col="green",lwd=6,main="After (Ref = green, Mov = blue)")
        lines(warped.tes.sig,typ="h",col="blue",lwd=2)
        par(mfrow=c(1,1))
      }

    }
  }

  if(printQ==TRUE) {
    print(paste("#Peaks (short) ref:",peak.count(ref.sig)))
    print(paste("#Peaks (short) tes:",peak.count(tes.sig)))
    print(paste("#Peaks warp:       ",peak.count(warped.tes.sig)))
    print(paste("DTW dist:                       ",algn$distance))
    print(paste("Aligned (short) corr score:     ",cs))
    print(paste("Test-warped (short)  corr score:",cstt))
    print(paste("Full sigs Aligned corr score:   ",csr))
    print(paste("Full test-warped corr score:    ",cst))
  }

  return(full.leng.warped.sig)

}


#------------------------------------------------
#Because the reference (plot with most peaks) will
#probably be different for each plot in a species group
#this routine dtw's the group, picking the proper
#reference observation for each plot.
#------------------------------------------------
align.peaks.for.group<-function(dmat.ini, break.ref.mat, init.normQ=F,plotQ=F,printQ=F) {

  #If there is to be normalizing, do it now:
  print(dim(dmat.ini))
  if(init.normQ==T) {
    dmat<-t(apply(dmat.ini,1,norm.profile))
    print(dim(dmat))
  } else {
    dmat<-dmat.ini
  }

  print(break.ref.mat)
  warped.group.mat<-NULL
  #Loop over plots:
  for(i in 1:nrow(break.ref.mat)) {
    plot.num<-break.ref.mat[i,1]
    st.idx<-break.ref.mat[i,2]
    sp.idx<-break.ref.mat[i,3]
    plot.mat<-dmat[,st.idx:sp.idx]

    #Pick out the reference obs vec for the plot
    ref.obs.idx<-break.ref.mat[i,4]
    plot.ref.vec<-plot.mat[ref.obs.idx,]
```

```
    tes.obs.idxs<-(1:nrow(dmat))[-ref.obs.idx] #so we know where to insert the results

    #Initialize a mat to hold the results for the plot and insert the reference (which doesn't
change)
    warped.plot.mat<-array(0,dim(plot.mat))
    warped.plot.mat[ref.obs.idx,]<-plot.ref.vec

    #Drop the ref vec from the plot mat and loop over the remaining columns
    plot.mat<-plot.mat[-ref.obs.idx,]
    for(j in 1:nrow(plot.mat)) {
      tes.obs.vec<-plot.mat[j,]
      print(paste("Ion plot#",plot.num," Obs#:", tes.obs.idxs[j]))

      #Check to make sure tes.obs.vec has peaks. If not, make a zero vec
      if((peak.count(tes.obs.vec)==0)) {
        #No peaks to align => make a zero vec
        warped.tes.obs.vec<-numeric(length(plot.ref.vec))
      } else {
        #Do the dtw if peaks present
        warped.tes.obs.vec<-align.peaks(tes.obs.vec,plot.ref.vec, normQ=F,plotQ,printQ)
      }

      warped.plot.mat[tes.obs.idxs[j],] <- warped.tes.obs.vec

      print("============================================")
    }

    warped.group.mat<-cbind(warped.group.mat,warped.plot.mat)

  }

  return(warped.group.mat)
}

#---------------------------------------------
#Count number of peaks in a vector/signal etc....
#---------------------------------------------
peak.count<-function(profl) {
  num.peak<-length(which(profl!=0))
  return(num.peak)
}

#------------------------------------------------------------
#Bin the peaks into C-size-categories (boiling point ranges).
#
# **NOTE** Bins are wrt our empirically observed e1618 even
#C-size-categories (i.e. C6,C8,C10,etc.) AND shifted wrt
#OUR INTERNAL STANDADARD!
#
#------------------------------------------------------------
bin.bp.range<-function(reten.times.wrt.intern.std) {

  C.cat.counts<-hist(reten.times.wrt.intern.std,breaks=C.bins,plot=F)$counts
  names(C.cat.counts)<-C.levels

  return(C.cat.counts)

}

#------------------------------------------------------------
#Compute contribution of SIC to total area mass of all SICs
#------------------------------------------------------------
sic.area.mass.contrib<-function(chromatogram.container){
        all.sum.sic.areas<-rep(0,5)
        for(i in 2:6){ #Assumes 5 SICs/chromatogram!
                sic.areas<-as.numeric(chromatogram.container[[i]][,2])
                #print(sic.areas)
                #print(is.na(sic.areas))
                #print("")
```

```
            if(length(sic.areas)==1 & (NA %in% sic.areas)) {
                sum.sic.areas<-0
            } else {
                sum.sic.areas<-sum(sic.areas)
            }
            all.sum.sic.areas[i-1]<-sum.sic.areas
    }

  #Contributions are in percentages and sum to 100:
    all.sim.contrib.perc.vec<-(all.sum.sic.areas/sum(all.sum.sic.areas)*100)
  names(all.sim.contrib.perc.vec)<-
c("Alkanes(%)","Cycloalkanes(%)","Aromatics(%)","Napthalenes(%)","Indanes(%)")

  return(all.sim.contrib.perc.vec)
}

#-----------------------------------------------------
#Categorize a vector of numbers into custom named
#intervals.
#E.g. for breaks = (0,1,10,87,100) and caregories = N, L, M, H
#breaks vec up into categories (0,1] (1,10], (10,87], (87,100)
#labeled N, L, M, H
#-----------------------------------------------------
custom.cut<-function(vec, breaks, categories){

  if(length(breaks) == (length(categories) + 1) ) {
    tmp<-cbind(vec, findInterval(vec, breaks))
    categ.mat<-data.frame(tmp,categories[tmp[,2]])

    return(categ.mat)

  } else {
    print("NUMBER OF CATEGORIES MUST BE 1-NUMBER OF BREAKS!")
    stop()
  }

}

#-----------------------------------------------------
#Try to see if the is an N-alkane pattern in the Alkane SIC
#-----------------------------------------------------
NAlkane.patternQ<-function(sic.mat, rt.tol, pk.hgh.tol.percent, num.nalk.peaks.tol) {

  #Get the height of the tallest peak in the alkane SIC.
  #It should be an N-alkane peak and other N-alkane peaks should
  #be at least ~pk.hgh.tol.percent*tallest.peak.hgh as tall:
  tallest.peak.hgh<-max(sic.mat[,2])
  #print(paste("Peak must be at least this high:", pk.hgh.tol.percent*tallest.peak.hgh/100, "to
count!"))

  #Loop over the positions of the E1618 N-alkanes:
  found.peak.info.mat<-NULL
  for(i in 1:length(C.bins[-c(1,17)])) {

    Cn.ref.peak<-C.bins[-c(1,17)][i]
    Cn.ref.peak.name<-C.levels[-c(16)][i]
    intevl <- (Cn.ref.peak + c(-rt.tol, rt.tol))

    #Look to see if any of the SIC alkane peaks are in the interval around the referernce peak:
    ind.vec<-sapply(1:length(sic.mat[,3]),function(x){sic.mat[x,3] >= intevl[1] & sic.mat[x,3] <=
intevl[2] })
    check.these.peak.idxs<-which(ind.vec==T)

    if(length(check.these.peak.idxs)>0) {

      tallest.peak.in.interv <- max(sic.mat[check.these.peak.idxs,2])
      tallest.peak.in.interv.idx <-
which(sic.mat[check.these.peak.idxs,2]==tallest.peak.in.interv)
      tallest.peak.in.interv.rt <- sic.mat[check.these.peak.idxs,3][tallest.peak.in.interv.idx]
```

```
      good.peakQ<-(tallest.peak.in.interv >= pk.hgh.tol.percent*tallest.peak.hgh/100)

      if(good.peakQ==TRUE) {
        found.peak.info.mat<-rbind(found.peak.info.mat, data.frame(tallest.peak.in.interv,
tallest.peak.in.interv.rt, Cn.ref.peak.name,Cn.ref.peak))
      }

    }

  }

  #Process any possible peaks of an N-alkane pattern a little more:
  enough.peaksQ<-FALSE            #Initialize these to FALSE in case no peak were found
  hgh.peak.in.found.peaksQ<-FALSE #and we don't make it into the if statement below
  if(!is.null(found.peak.info.mat)){

    colnames(found.peak.info.mat)<-c("Peak Height","(Shifted) R.T.", "C-N", "Ref R.T.")
    rownames(found.peak.info.mat)<-NULL

    #See if we found more than one lucky peak:
    if(nrow(found.peak.info.mat)>=num.nalk.peaks.tol){
      enough.peaksQ <- TRUE
    } else {
      enough.peaksQ <- FALSE
    }
    #Check to see that the tallest peak in the alkane SIC is in the set of peaks we found.
    if(tallest.peak.hgh %in% found.peak.info.mat[,1]) {
      hgh.peak.in.found.peaksQ<-TRUE
    } else {
      hgh.peak.in.found.peaksQ<-FALSE
    }

  }
  #print(found.peak.info.mat)

  #T is yes, there is a detected N-alkane pattern, F is otherwise:
  passQ <- (enough.peaksQ & hgh.peak.in.found.peaksQ)

  return(passQ)
}


#----------------------------------------------------
#See if there is hash and trash in the Alkanes SIC
#This should work for any SIC however since it just counts peaks
#----------------------------------------------------
hash.and.trashQ<-function(alk.mat,num.peaks.tol) {

  #Should we remove peaks we think are Cn-s first??

  num.peaks<-nrow(alk.mat)
  if(num.peaks>num.peaks.tol){
    hnt<-TRUE
  } else {
    hnt<-FALSE
  }

  return(hnt)

}


#-------------------------------------------------------------
#
#-------------------------------------------------------------
make.fv<-function(chromatogram, percent.breaks, break.categories){
```

```
  fv1<-as.character(custom.cut(sic.area.mass.contrib(chromatogram), percent.breaks,
break.categories)[,3])

  #Examine the Alkanes for patterns:
  if((fv1[1]=="High") | (fv1[1]=="Med") | (fv1[1]=="Low") ) { #Look first to see if there is an
Alkane SIC

    alkanes.sic <- chromatogram[[2]]

    #Check for an N-alkane pattern:
    nalk.patternQ <- NAlkane.patternQ(alkanes.sic, rt.tol=0.18, pk.hgh.tol.percent=1,
num.nalk.peaks.tol=3)
    #If an N-alkane pattern is detected check for hash and trash:
    if(nalk.patternQ == TRUE) {
      hash.trashQ <- hash.and.trashQ(alkanes.sic,num.peaks.tol=6)
    } else { #If there are Alkanes but no N-alkane pattern, don't call peaks hash and trash
      hash.trashQ <- FALSE
    }

  } else { #If no Alkane SIC in chromatogram => set nalk.patternQ to FALSE
    nalk.patternQ <- FALSE
    hash.trashQ <- FALSE
  }

  #dat<-data.frame(NalkQ.vec,htQ.vec, dat)
  fv2 <- data.frame(nalk.patternQ, hash.trashQ, fv1[1], fv1[2], fv1[3], fv1[4], fv1[5])
  colnames(fv2)<-c("Nalk","Hash","Alk","Cyclo","Aro","Nap","Ind")

  return(fv2)

}
```

process.chromatograms.utils.R: Utility/helper functions for the routines above.

```
#--------------------------------------------------
#Plot our E1618 codification over a chromatogram.
#plot.mat may be a TIC or any SIC
#--------------------------------------------------
overlay.e1618.reference<-function(plot.mat) {

  if(names(dev.cur())=="null device") {
    #print("IM HERE!")
    dev.new()
  }
  dev.off() #Shut off the screen if it was being used and had been parsed.

  rt.lims<-c(min(c(C.bins[c(-1,-17)],plot.mat[,3])), max(c(C.bins[c(-1,-17)],plot.mat[,3])))
  #Set to 10% higher than highest peak in the chromatogram:
  ab.lims<-c(0,max(plot.mat[,2])+(0.1*max(plot.mat[,2])))

  plot(C.bins[c(-1,-17)], rep(ab.lims[2],15), xlim=rt.lims, ylim=ab.lims, col="green", lwd=3,
typ="h",ylab="",xlab="")
  text(C.bins[c(-1,-17)], rep(ab.lims[2],15), labels=C.levels[-16])
  par(new=T)
  plot(plot.mat[,3], plot.mat[,2],typ="h",xlim=rt.lims, ylim=ab.lims)

}


#--------------------------------------------------
#Plot our E1618 codification over a chromatogram.
#plot.mat may be a TIC or any SIC
#--------------------------------------------------
overlay.e1618.reference2<-function(plot.mat, overlayQ=TRUE) {

  if(names(dev.cur())=="null device") {
    #print("IM HERE!")
    dev.new()
  }
  dev.off() #Shut off the screen if it was being used and had been parsed.

  rt.lims<-c(min(c(C.bins[c(-1,-17)],plot.mat[,3])), max(c(C.bins[c(-1,-17)],plot.mat[,3])))
  #Set to 10% higher than highest peak in the chromatogram:
  ab.lims<-c(0,max(plot.mat[,2])+(0.1*max(plot.mat[,2])))

  plot(plot.mat[,3], plot.mat[,2],typ="h",xlim=rt.lims, ylim=ab.lims, ylab="Normalized Area",
xlab="Shifted R.T. (min)")
  if(overlayQ==TRUE){
    par(new=T)
    plot(C.bins[c(-1,-17)], rep(ab.lims[2],15), xlim=rt.lims, ylim=ab.lims, col="green", lwd=3,
typ="h",ylab="",xlab="")
    text(C.bins[c(-1,-17)], rep(ab.lims[2],15), labels=C.levels[-16])
  }

}


#------------------------------------------------------------
#Compute contribution of SIC to total area mass of all SICs
#------------------------------------------------------------
Cn.area.mass.contrib<-function(chromatogram.container){
  #all.sum.sic.areas<-rep(0,5)
  #Loop over SICs calculating each Cn's mass contribution as a percentage
  #within the SIC and over the totality og the SICs:
  sic.Cn.mass.contribs<-array(NA,c(5,length(C.levels))) #Assumes 5 SICs/chromatogram!
  for(i in 2:6){ #Assumes 5 SICs/chromatogram!

    sic.shifted.rt<-chromatogram.container[[i]][,3] #Shifted retension times for a SIC
```

```
    sic.peak.hgths<-chromatogram.container[[i]][,2]  #Peak heights for a SIC
    sic.Cn.counts<-bin.bp.range(sic.shifted.rt)

    sic.Cn.info.mat<-data.frame(sic.shifted.rt, sic.peak.hgths, cut(sic.shifted.rt,C.bins),
C.levels[as.numeric(cut(sic.shifted.rt,C.bins))])
    sic.Cn.mass.contribs[i-1,]<-
sapply(1:length(C.levels),function(x){sum(sic.Cn.info.mat[which(sic.Cn.info.mat[,4]==C.levels[x])
,2])})
    #print(sic.Cn.info.mat)
    #print(tmp)
    #print("")
  }
  colnames(sic.Cn.mass.contribs)<-C.levels
  rownames(sic.Cn.mass.contribs)<-c("Alkanes","Cycloalkanes","Aromatics","Napthalenes","Indanes")
  #print(sic.Cn.mass.contribs)
  #print(rowSums(sic.Cn.mass.contribs))
  #print(sic.Cn.mass.contribs/rowSums(sic.Cn.mass.contribs) * 100)
  within.sic.percent.contribs<-(sic.Cn.mass.contribs/rowSums(sic.Cn.mass.contribs) * 100)
  #print(is.nan(within.sic.percent.contribs))
  within.sic.percent.contribs[which(is.nan(within.sic.percent.contribs)==TRUE,arr.ind=TRUE)]<-0
  #print(within.sic.percent.contribs)

  totalCn.percent.contribs<-(colSums(sic.Cn.mass.contribs)/sum(sic.Cn.mass.contribs) *100)
  #print(totalCn.percent.contribs)

  Cn.mass.percent.contrib.info.list<-list(within.sic.percent.contribs,totalCn.percent.contribs)
  names(Cn.mass.percent.contrib.info.list)<-c("Cn mass %-contrib by SIC", "Cn mass %-contrib
across SICs")

  #Now determine Cn mass %-contribs for whole chromatogrms
  shifted.rt<-chromatogram.container[[1]][,3] #Shifted retension times for chromatogram
  peak.hgths<-chromatogram.container[[1]][,2]  #Peak heights for chromatogram
  Cn.counts<-bin.bp.range(shifted.rt)

  Cn.info.mat<-data.frame(shifted.rt, peak.hgths, cut(shifted.rt,C.bins),
C.levels[as.numeric(cut(shifted.rt,C.bins))])
  total.Cn.mass.contribs.whole.chrom<-
sapply(1:length(C.levels),function(x){sum(Cn.info.mat[which(Cn.info.mat[,4]==C.levels[x]),2])})
  names(total.Cn.mass.contribs.whole.chrom)<-C.levels
  #print(Cn.info.mat)
  #print(total.Cn.mass.contribs)
  total.Cn.mass.contribs.whole.chrom.perc<-
(total.Cn.mass.contribs.whole.chrom/sum(total.Cn.mass.contribs.whole.chrom) *100)
  #print(total.Cn.mass.contribs.whole.chrom.perc)
  #print(sum(total.Cn.mass.contribs.whole.chrom.perc))

  Cn.mass.percent.contrib.info.list<-list(within.sic.percent.contribs,totalCn.percent.contribs,
total.Cn.mass.contribs.whole.chrom.perc)
  names(Cn.mass.percent.contrib.info.list)<-c("Cn mass %-contrib by SIC", "Cn mass %-contrib
across SICs", "Total Cn mass %-contrib in chromatogram")

  return(Cn.mass.percent.contrib.info.list)
}


#------------------------------------------------------
#Classify Chromatogram as Light, Medium or Heavy
#light, most mass between C4-C9
#medium, most mass between C8-C13
#heavy, most most mass between C9-C20+
#------------------------------------------------------
classify.mass<-function(mass.percent.contrib.vec, printQ=FALSE){

  c4c9<-sum(mass.percent.contrib.vec[1:4])
  c8c13<-sum(mass.percent.contrib.vec[3:8])
  c9c20p<-sum(mass.percent.contrib.vec[4:16])

  c14c20p<-sum(mass.percent.contrib.vec[9:16])  #Clarify if really is heavy, or is there just a
little bit of mass past C13
```

```
  c15c20p<-sum(mass.percent.contrib.vec[10:16]) #Helps to see if it's a weathered

  mass.class<-"NULL"
  weathered.indic<-"No"

  if(printQ==TRUE){
    print(paste("C4-C9 %-mass:",c4c9))
    print(paste("C8-C13 %-mass:",c8c13))
    print(paste("C9-C20+ %-mass:",c9c20p))
  }

  idx.most<-which(c(c4c9,c8c13,c9c20p)==max(c(c4c9,c8c13,c9c20p)))[1] #Pick the first if there
are ties
  if(idx.most==1){
    mass.class<-"Light"
  }
  if(idx.most==2){
    mass.class<-"Medium"
  }
#   if(idx.most==3){
#     mass.class<-"Heavy"
#   }
  if(idx.most==3 & c14c20p>=5){
    mass.class<-"Heavy"
  }
  if(idx.most==3 & c14c20p<5){ #Define a "little bit of mass past C13 as <12%
    mass.class<-"Medium"
  }

  if(c4c9==0 & c8c13==0 & c15c20p>0){
    weathered.indic<-"May be weathered"
  }

  mass.info.vec <- c(mass.class,weathered.indic)
  names(mass.info.vec)<-c("Mass Class","Weathered?")

  return(mass.info.vec)

}


#---------------------------------------------------------
#Test for an N-alkane pattern in a set of peaks
#---------------------------------------------------------
NAlkane.patternQ<-function(sic.mat, rt.tol, pk.hgh.tol.percent, num.nalk.peaks.tol) {

  #Get the height of the tallest peak in the alkane SIC.
  #It should be an N-alkane peak and other N-alkane peaks should
  #be at least ~pk.hgh.tol.percent*tallest.peak.hgh as tall:
  tallest.peak.hgh<-max(sic.mat[,2])
  #print(paste("Peak must be at least this high:", pk.hgh.tol.percent*tallest.peak.hgh/100, "to
count!"))

  #Loop over the positions of the E1618 N-alkanes:
  found.peak.info.mat<-NULL
  for(i in 1:length(C.bins[-c(1,17)])) {

    Cn.ref.peak<-C.bins[-c(1,17)][i]
    Cn.ref.peak.name<-C.levels[-c(16)][i]
    intevl <- (Cn.ref.peak + c(-rt.tol, rt.tol))

    #Look to see if any of the SIC alkane peaks are in the interval around the referernce peak:
    ind.vec<-sapply(1:length(sic.mat[,3]),function(x){sic.mat[x,3] >= intevl[1] & sic.mat[x,3] <=
intevl[2] })
    check.these.peak.idxs<-which(ind.vec==T)

    if(length(check.these.peak.idxs)>0) {
```

```
      tallest.peak.in.interv <- max(sic.mat[check.these.peak.idxs,2])
      tallest.peak.in.interv.idx <-
which(sic.mat[check.these.peak.idxs,2]==tallest.peak.in.interv)
      tallest.peak.in.interv.rt <- sic.mat[check.these.peak.idxs,3][tallest.peak.in.interv.idx]

      good.peakQ<-(tallest.peak.in.interv >= pk.hgh.tol.percent*tallest.peak.hgh/100)

      if(good.peakQ==TRUE) {
        found.peak.info.mat<-rbind(found.peak.info.mat, data.frame(tallest.peak.in.interv,
tallest.peak.in.interv.rt, Cn.ref.peak.name,Cn.ref.peak))
      }

    }

  }

  #Process any possible peaks of an N-alkane pattern a little more:
  enough.peaksQ<-FALSE            #Initialize these to FALSE in case no peak were found
  hgh.peak.in.found.peaksQ<-FALSE #and we don't make it into the if statement below
  if(!is.null(found.peak.info.mat)){

    colnames(found.peak.info.mat)<-c("Peak Height","(Shifted) R.T.", "C-N", "Ref R.T.")
    rownames(found.peak.info.mat)<-NULL

    #See if we found more than one lucky peak:
    if(nrow(found.peak.info.mat)>=num.nalk.peaks.tol){
      enough.peaksQ <- TRUE
    } else {
      enough.peaksQ <- FALSE
    }
    #Check to see that the tallest peak in the alkane SIC is in the set of peaks we found.
    if(tallest.peak.hgh %in% found.peak.info.mat[,1]) {
      hgh.peak.in.found.peaksQ<-TRUE
    } else {
      hgh.peak.in.found.peaksQ<-FALSE
    }

  }
  #print(found.peak.info.mat)

  #T is yes, there is a detected N-alkane pattern, F is otherwise:
  passQ <- (enough.peaksQ & hgh.peak.in.found.peaksQ)

  return(passQ)
}
```

categorize_chromatograms.R: Functions and wrappers for the fitting/application of Bayes nets and Markov random fields to the category feature vectors derived from the chromatogram data.

```
#-----------------------------------------------------
#Categorize a chromatogram with a supplied Bayes Net
#-----------------------------------------------------
categorize.chromatogram<-function(supplied.bn, explanitory.observations, plotQ=FALSE){

  bn.with.ev <- setFinding(supplied.bn, nodes=names(explanitory.observations),
states=explanitory.observations)
  pred.probs<-querygrain(bn.with.ev,nodes=c("Class"), type="conditional")

  if(plotQ==TRUE){
    #plot(1:length(names(pred.probs)),pred.probs*100,
typ="h",xaxt="n",xlab="",ylab="Pr(%)",main="E1618 Class Probabilities",lwd=6)

plot(1:length(names(pred.probs)),pred.probs*100,typ="h",xaxt="n",xlab="",ylab="Pr(%)",main="E1618
Class Probabilities",lwd=6)
    #axis(1, at=1:length(names(pred.probs)),labels=names(pred.probs), col.axis="red", las=2)
    axis(1, at=1:length(names(pred.probs)),labels=FALSE, las=2)
```

```
    text(1:length(names(pred.probs)),rep(-6,length(names(pred.probs))), labels = 
names(pred.probs), srt = 20, xpd = T, cex=1.2 ,col="red")
  }

  return(pred.probs)

}

#-----------------------------------------------------
#Classify an input chromatogram according the E1618
#scheme
#-----------------------------------------------------
E1618.class.chromatogram<-function(supplied.bn, formated.chromatogram, mass.class.type="all", 
percent.breaks, break.categories, plotQ=FALSE, printQ=FALSE){

  #Make the chromatogram into a feature vector:
  feature.vector<-make.fv(formated.chromatogram, percent.breaks, break.categories)

  #Use the supplied bayes net of MRF to find E1618 class probs for the chromatogram:
  e1618.categ.prob.vec <- categorize.chromatogram(supplied.bn, feature.vector, plotQ)*100

  #Classify the mass distribution:
  if(mass.class.type=="all"){ #Use whole chromatogram to classify the mass distribution
    mass.vec<-Cn.area.mass.contrib(formated.chromatogram)[[3]]
  }
  if(mass.class.type=="EIC"){ #Only use the extracted ion chromatograms to classify the mass
    mass.vec<-Cn.area.mass.contrib(formated.chromatogram)[[2]]
  }

  mass.info<-classify.mass(mass.vec,printQ)

  return(list(mass.info,e1618.categ.prob.vec))

}
```

bn.performance.utils.R: Function to asses the performance of a Bayes net or Markov random field on a set of categorized chromatogram feature vectors. The core of the algorithm is hold-one-out cross-validation.

```
#------------------------------------------------
#Run performance metrics on category feature
#vector matrix, on a fit Bayes net or Markov
#random field
#------------------------------------------------
e1618.bn.performance.iter <- function(dmat, bn.fit) {

  e1618.classes<-levels(dmat[,ncol(dmat)])

  performance.mat<-as.data.frame(array(NA,c(length(e1618.classes), 5 )))

  colnames(performance.mat)<-c("Class","First.Choice(%)","Top2(%)","Top3(%)","Not.In.Top3(%)")

  for(class.idxxxx in 1:length(e1618.classes)) {
    cls<-e1618.classes[class.idxxxx]

    class.idxs<-which(dmat[,ncol(dmat)]==cls) #These should be obs indices in a class!!!!!!
    print(class.idxs)

    score.mat<-array(0,c(length(class.idxs),9))
    colnames(score.mat)<-c("idx","Aromatic","Gasoline","Isoparaffinic","Misc","N-Alkane","Nap-
Par","Oxygenated","PD")
    top3.class.names.mat<-array("x",c(length(class.idxs),3))
    #print(head(top3.class.names.mat))
    top3.class.probs.mat<-array(0,c(length(class.idxs),3))

    for(i in 1:length(class.idxs)){
```

```
      idx<-class.idxs[i]
      #chro.idx<-as.numeric(rownames(dat2[idx,])) #Correct the index for ilrc DB accounting for
dropped (bad) chromatograms
      #print(paste("Chromatogram#:",chro.idx,"Class:",dat2[idx,ncol(dat2)], ". And
again:",chro.info[chro.idx,4]))

      expl.vars<-dmat[idx,-ncol(dmat)] #Drop the labels
      #plot.gc.info2(ilrc.container[[chro.idx]], chro.info[chro.idx,2], chro.info[chro.idx,4],
axis.typ="shifted")
      #score.vec<-c(idx,round(categorize.chromatogram(bn.fit, expl.vars, plotQ=FALSE)*100,3))
      #print(score.vec)
      #score.mat<-rbind(score.mat,score.vec)
      score.mat[i,]<-c(idx,round(categorize.chromatogram(bn.fit, expl.vars, plotQ=FALSE)*100,3))
      #print("")

      sv<-score.mat[i,-1]
      top3.vec<-sv[order(sv,decreasing=T)][1:3]
      #print(top3.vec)
      nmes<-names(top3.vec)
      probs<-as.numeric(top3.vec)

      #top3.class.names.mat<-rbind(top3.class.names.mat,nmes)
      #print(nmes)
      top3.class.names.mat[i,]<-nmes
      #top3.class.probs.mat<-rbind(top3.class.probs.mat,probs)
      top3.class.probs.mat[i,]<-probs
    }
    #score.mat
    #top3.class.names.mat
    #top3.class.probs.mat

    num1<-sum(top3.class.names.mat[,1]==cls)
    num2<-sum(top3.class.names.mat[,2]==cls)
    num3<-sum(top3.class.names.mat[,3]==cls)
    #
    first.choice.perc<-round(num1/nrow(top3.class.names.mat)*100,3)                        #%
First choice
    top.two.perc<-round((num1+num2)/nrow(top3.class.names.mat)*100,3)                      #% Top
2
    top.three.perc<-round((num1+num2+num3)/nrow(top3.class.names.mat)*100,3)               #% Top
3
    not.in.top.three.perc<-round(100-((num1+num2+num3)/nrow(top3.class.names.mat)*100),3) #% Not
in top 3
    performance.vec<-data.frame(as.character(cls), first.choice.perc, top.two.perc,
top.three.perc, not.in.top.three.perc)
    #print(performance.vec)
    performance.mat[class.idxxxx,]<-performance.vec
  }

  return(list(performance.mat,top3.class.names.mat,top3.class.probs.mat))

}
```

gui_utils.R: Functions to implement a graphical user interface (GUI) for displaying the chromatogram data and using a fit Bayes net or Markov random field to estimate E1618 class membership beliefs.

```
require(gWidgets)
options(guiToolkit="RGtk2")
#require(gWidgets2)
require(gWidgetsRGtk2)  #For some reason ggraphics() is throwing an error from
gWidget2RGtk2....... Be aware. Probably will be fixes in an updated version of the package
                        #Substituting gWidgetsRGtk2
```

```
#require(gWidgets2RGtk2)

#NOTE: This version only works with RGtk2 so will be a problem for R>3.0.0!!!!!

#----------------------------------------------------------
#Try to use Verzani's design pattern under gWidgets. Seems more modular
#and modifyable than his manupulate
#Reference: gWidgets Vignette: Examples for gWidgets John Verzani
#http://cran.r-project.org/web/packages/gWidgets/vignettes/gWidgets.pdf
#----------------------------------------------------------
viewer.gui<-function(chromatogram.container){

  #First define the spans of chromatograms/plots we want to look through
  ## set up
  chromatogram.nums <- 1:length(chromatogram.container)
  plot.nums <- 1:6

  #Handler to update the graphical output
  updatePlot <- function(h,...) {
    plot.mat<-chromatogram.container[[svalue(obs.num)]][[svalue(plot.num)]]
    plot.nme<-names(chromatogram.container[[svalue(obs.num)]])[svalue(plot.num)]

    if(!is.na(plot.mat)[1]){ #Plot the TIC or EIC (Extracted Ion Chromatogram) if something is
there
      #Span of shifted R.T.s:
      rt.lims<-c(min(c(C.bins[c(-1,-17)],plot.mat[,3])), max(c(C.bins[c(-1,-17)],plot.mat[,3])))
      #Bar (Area) height range with room made for Cn-series lettering:
      ab.lims<-c(0,max(plot.mat[,2])+(0.1*max(plot.mat[,2])))

      #If checkbox is checked for superimposing the Cn-series:
      if(svalue(cb)){
        plot(C.bins[c(-1,-17)], rep(ab.lims[2],15), xlim=rt.lims, ylim=ab.lims, col="green",
lwd=3, typ="h",ylab="",xlab="")
        text(C.bins[c(-1,-17)], rep(ab.lims[2],15), labels=C.levels[-16])
        par(new=T) #A necessary jerry-rig to get the green bars to plot under the chromatogram
AND the chromatogram to plot w/o the green lines too
      }
      #Plot the selected chromatogram:
      plot(plot.mat[,3], plot.mat[,2],typ="h",xlim=rt.lims, ylim=ab.lims, xlab="Shifted R.T.",
ylab="Normalized Area", main=paste("Plot#:",svalue(plot.num),"  ",plot.nme))

    } else {
      #If nothing is in the plot (i.e. no ions, or a (mistaken) absence of a TIC plot this:
      plot(c(0,1),c(1,1),typ="l",axes=F,xlab="",ylab="",main=paste("No
plot#:",svalue(plot.num),plot.nme))
    }
  }


  #Handler for generating class probs:
  #Perhaps later add these in as combo-boxes:
  percent.breaks <- c(0,1,10,30,101)
  break.categories <- c("None","Low","Med","High")

  genPred<-function(h, ...){

    feat.vec<-make.fv(chromatogram.container[[svalue(obs.num)]], percent.breaks=percent.breaks,
break.categories = break.categories)
    print(feat.vec)

    #tmph<-ggraphics(cont=BigGroup2,dpi=20)
    #visible(tmph)<-T
    pred.categ.probs<-round(categorize.chromatogram(bn.aic, feat.vec, plotQ=T)*100,3)
    print(pred.categ.probs)

    #gg1<-ggraphics(cont=notebook, expand = TRUE, label="XXXX")
    #chartSeries(get(symbol),subset="last 2 months", name="Trends")
    #junk2<-round(categorize.chromatogram(bn.aic, fv2, plotQ=TRUE)*100,3)
    #visible(gg1) <- T
```

```
  } #end genPred handler

 #Define the widgets:
 #For Tab 1:
 obs.num <- gcombobox(chromatogram.nums, handler=updatePlot)
 plot.num <- gcombobox(plot.nums, handler=updatePlot)
 cb <- gcheckbox(text="Overlay Cn",checked=FALSE,handler=updatePlot)
 refr <- gbutton(text="Refresh", handler=updatePlot)
 #For Tab 2:
 pred.net <- gbutton(text="Pred Probs", handler=genPred)

 #The layout:
 window <- gwindow("E1618 Classifier GUI", visible=F)
 #notebook <- gnotebook(cont = window)

 #Tab 1: The chromatogram plots
 BigGroup <- ggroup(cont=window, label="Chromatograms")
 #BigGroup <- ggroup(cont=notebook, label="Chromatograms")
 group <- ggroup(horizontal=FALSE, container=BigGroup)
 tmp <- gframe("Chro Num:", container=group)
 add(tmp, obs.num)

 tmp <- gframe("Plot num:", container=group)
 add(tmp,plot.num)
 tmp <- gframe("Overlay Cn?", container=group)
 add(tmp,cb)
 tmp <- gframe("Refresh Plot?", container=group)
 add(tmp,refr)

 tmp <- gframe("Predict", container=group)
 add(tmp, pred.net)

 tmp<-ggraphics(cont=BigGroup)
 visible(tmp)<-T
 #add(BigGroup, tmp)

 #Tab 2: The predictions
 #BigGroup2 <- ggroup(cont=notebook, label="Predictions")
 #group2 <- ggroup(horizontal=FALSE, container=BigGroup2)
 #tmp <- gframe("Predict", container=group2)
 #add(tmp, pred.net)

 #??glayout, glabel
 #group3 <- ggroup(horizontal=FALSE, container=BigGroup2)
 #addSpring(group3)
 #tmp <- glabel("HERE", container=group3)

 #tmp2<-ggraphics(cont=BigGroup2)
 #visible(tmp2)<-T
 #add(BigGroup2, ggraphics())

 #svalue(notebook)<-2
 visible(window)<-T
}
```

Appendix 2: Expert system results for unknown samples

Table A-1: Challenge Samples Results

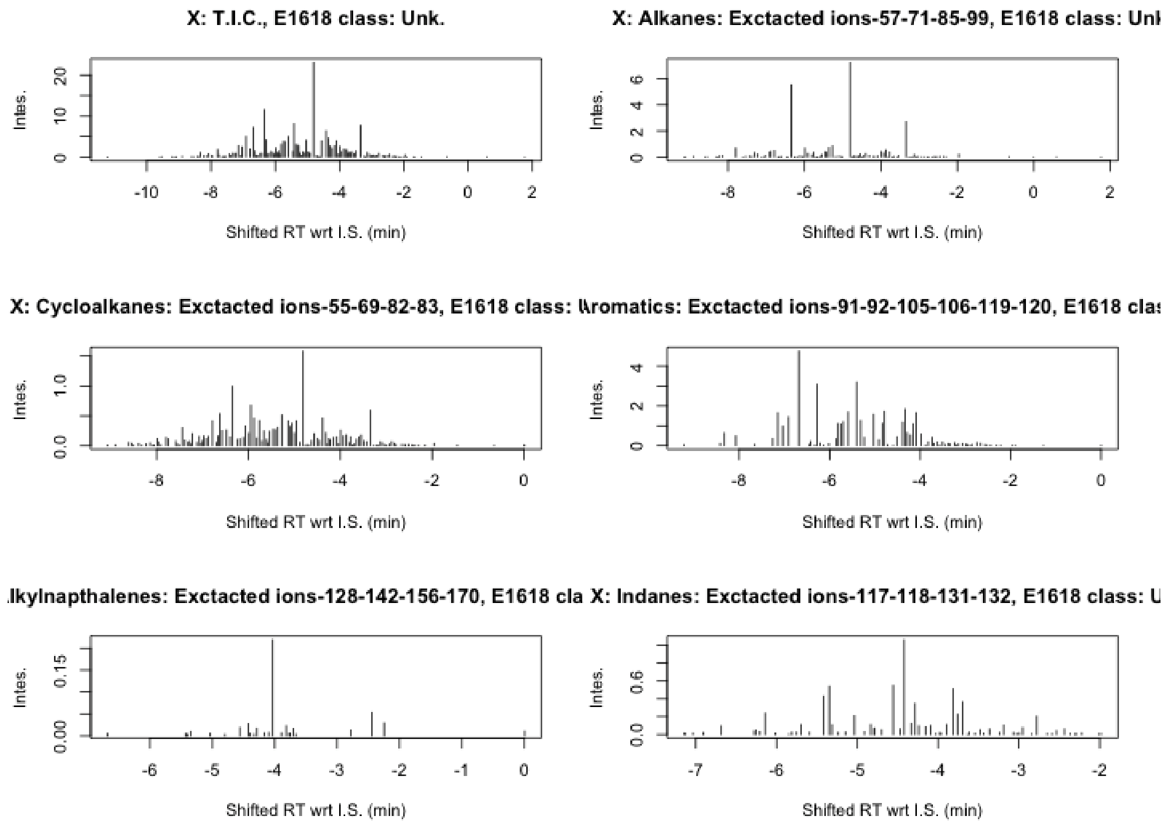
Substrate burned to 50% mass, then spiked with Ignitable Liquid				
File Name	**Ignitable Liquid**	**Class**	**Substrate**	**Expert System ID (top 3)**
Blc 54	Crown Paint Thinner	MPD	blue carpet	MOxy, MPD
Blc 59	Crown Lacquer Thinner	MOxy	blue carpet	Cannot ID
Blc 66	Kingsford lighter fluid	MPD	blue carpet	MOxy, MPD
Blc Blk	unburned blank		blue carpet	Cannot ID
Blc E85	E85	Gasoline	blue carpet	MOxy, MD
Blc Gas	Gasoline	Gasoline	blue carpet	MMisc
Blc Ker	Kerosene	HPD	blue carpet	HPD, HMisc, HOxy
Blc MtdB	burned blank		blue carpet	HMisc, HOxy
Brc 54	Crown Paint Thinner	MPD	brown carpet	MPD, MOxy
Brc 59	Crown Lacquer Thinner	MOxy	brown carpet	HOxy, No ID
Brc 66	Kingsford lighter fluid	MPD	brown carpet	MOxy, MPD
Brc Blk	unburned blank		brown carpet	Cannot ID
Brc E85	E85	Gasoline	brown carpet	MMisc
Brc Gas	Gasoline	Gasoline	brown carpet	MMisc, MPD
Brc Ker	Kerosene	HPD	brown carpet	HPD, HMisc, HOxy
Brc MtdB	burned blank		brown carpet	Cannot ID
	Crown Paint Thinner			MNap-Par, MPD, MOxy

CP 54		MPD	carpet pad	
CP 59	Crown Lacquer Thinner	Oxy	carpet pad	Cannot ID
CP 66	Kingsford lighter fluid	MPD	carpet pad	MOxy, MPD
CP Blk	unburned blank		carpet pad	Cannot ID
CP E85	E85	Gasoline	carpet pad	MOxy
CP Gas	Gasoline	Gasoline	carpet pad	MMisc
CP Ker	Kerosene	HPD	carpet pad	HPD, HMisc, Hoxy
CP MtdB	burned blank		carpet pad	Cannot ID
OO 54	Crown Paint Thinner	MPD	old oak	MNap-Par, MOxy, MPD
OO 59	Crown Lacquer Thinner	Oxygenate	old oak	HOxy
OO 66	Kingsford lighter fluid	MPD	old oak	MPD
OO Blk	unburned blank		old oak	Cannot ID
OO E85	E85	Gasoline	old oak	MMisc
OO Gas	Gasoline	Gasoline	old oak	MMisc, MOxy, MPD
OO Ker	Kerosene	HPD	old oak	HPD, HMisc, HOxy
OO MtdB	burned blank		old oak	HPD
PW 54	Crown Paint Thinner	MPD	Plywood	MPD, MOxy, MMisc
PW 59	Crown Lacquer Thinner	Oxygenate	Plywood	HOxy
PW 66	Kingsford lighter fluid	MPD	Plywood	MMisc
PW Blk	unburned blank		Plywood	Cannot ID

PW E85	E85	Gasoline	Plywood	HOxy
PW Gas	Gasoline	Gasoline	Plywood	MMisc
PW Ker	Kerosene	HPD	Plywood	HPD, HMisc, HOxy
PW MtdB	burned blank		Plywood	HOxy
YP 54	Crown Paint Thinner	MPD	Yellow Pine	MMisc, MPD
YP 59	C	Oxygenate	Yellow Pine	Cannot ID
YP 66	Kingsford lighter fluid	MPD	Yellow Pine	MMisc, MOxy
YP Blk	unburned blank		Yellow Pine	MMisc, Gasoline
YP E85	E85	Gasoline	Yellow Pine	MMisc, MOxy
YP Gas	Gasoline	Gasoline	Yellow Pine	MMisc
YP Ker	Kerosene	HPD	Yellow Pine	HPD
YP MtdB	burned blank		Yellow Pine	Cannot ID
	Substrate spiked with IL then ignited			
GasYP	Gasoline	Gasoline	Yellow Pine	MMisc
KerYP	Kerosene	HPD	Yellow Pine	Cannot ID
	IL spiked on unburned substrate			
UBSBLC54	Crown Paint Thinner	MPD	Unburned blue carpet	MNap-Par/MOxy, MPD
UBSBLC59	Crown Lacquer Thinner	MOxy	Unburned blue carpet	Cannot ID
UBSYP54	Crown Paint Thinner	MPD	Unburned Yellow	MOxy, MPD, MMisc

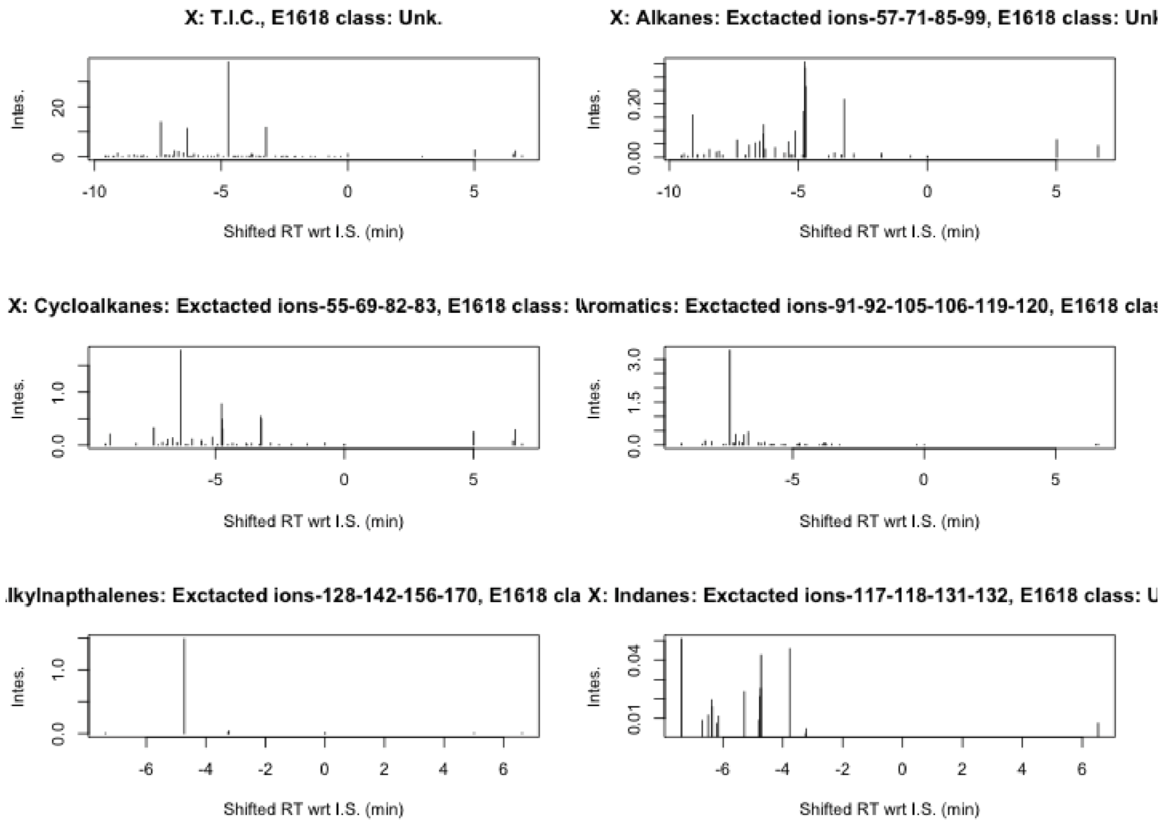
			Pine	

Unknown #1: 54YP, Crown Paint Thinner on Yellow Pine; MPD class



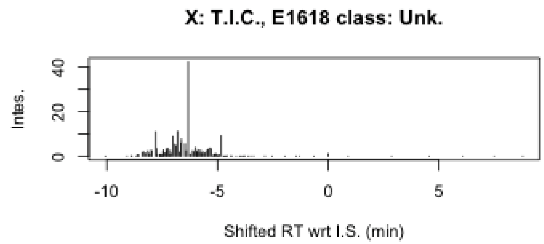
Unknown #	Pred Classes	
1	Med Misc	
	Med Misc	MPD

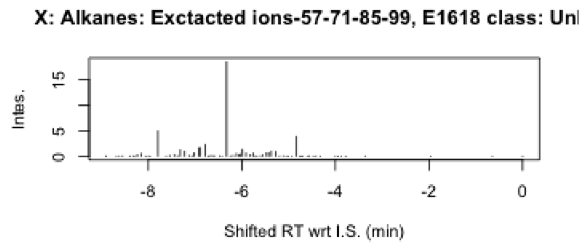
Unknown 2: 59 YP, Crown Lacquer Thinner on Yellow Pine; Medium MISC

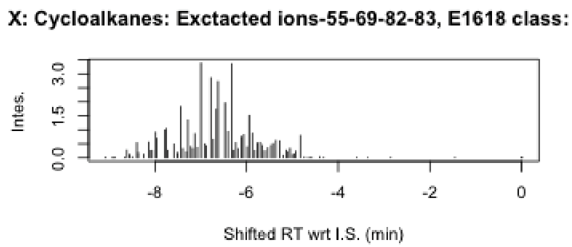


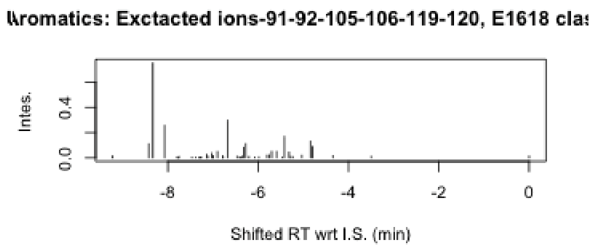
Unknown #	**Pred Classes**
2	CAN NOT ID

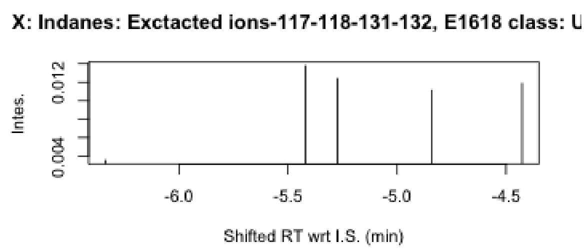
Unknown 3: 66yp, Kingsford Charcoal Lighter on Yellow Pine, MDP





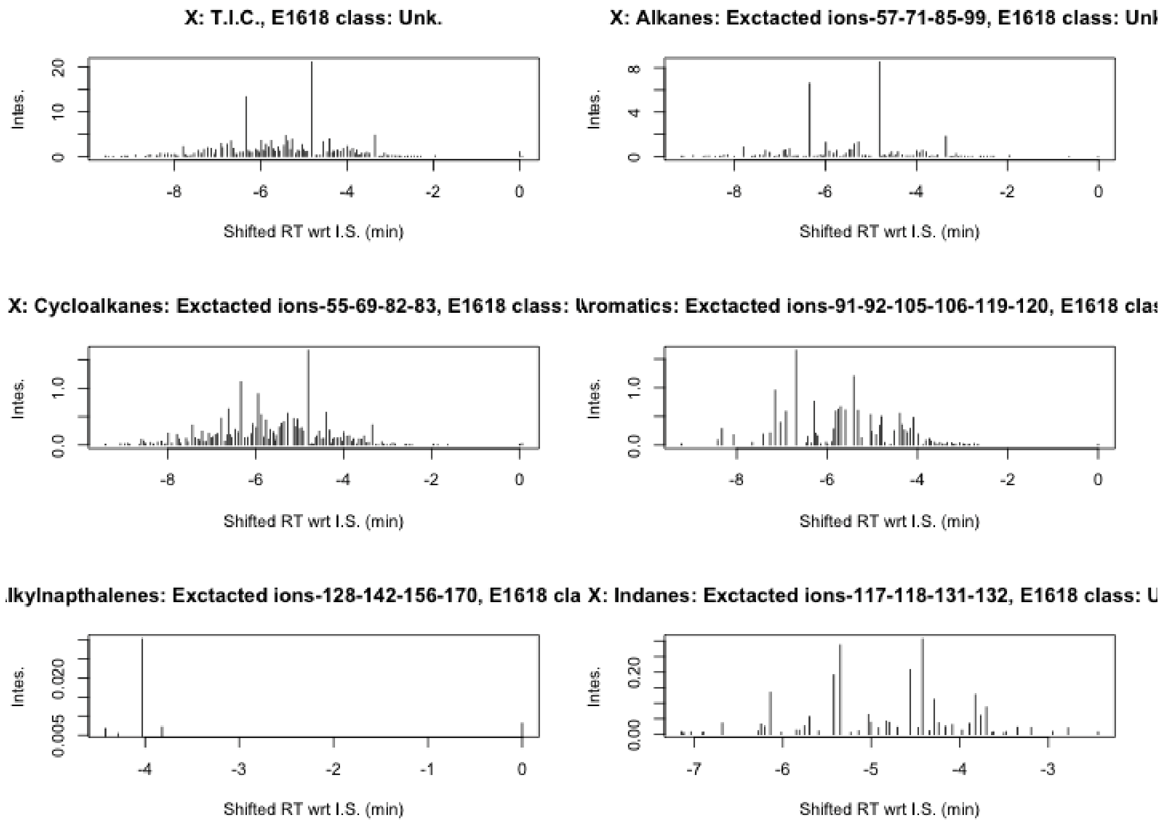






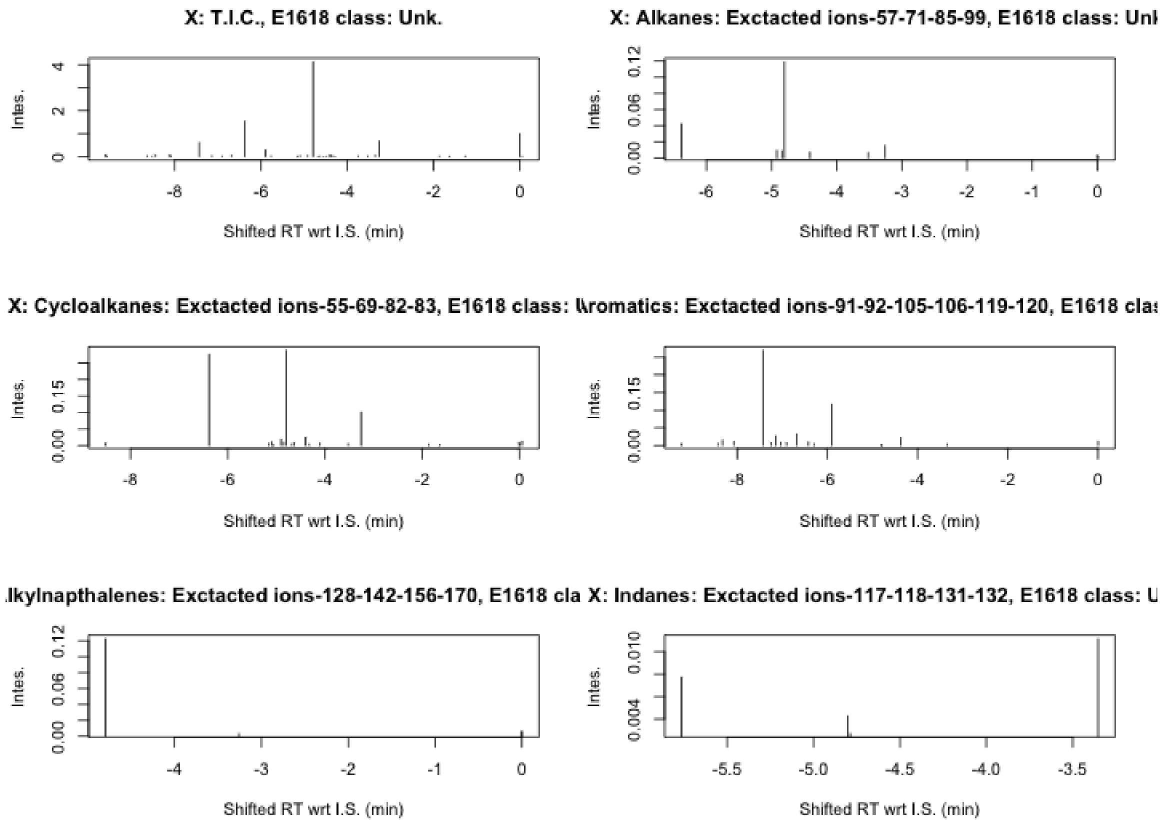
Unknown #	Pred Classes	
3	M Oxy	MPD

Unknown 4: blc54; Crown Paint Thinner on Blue Carpet; MPD



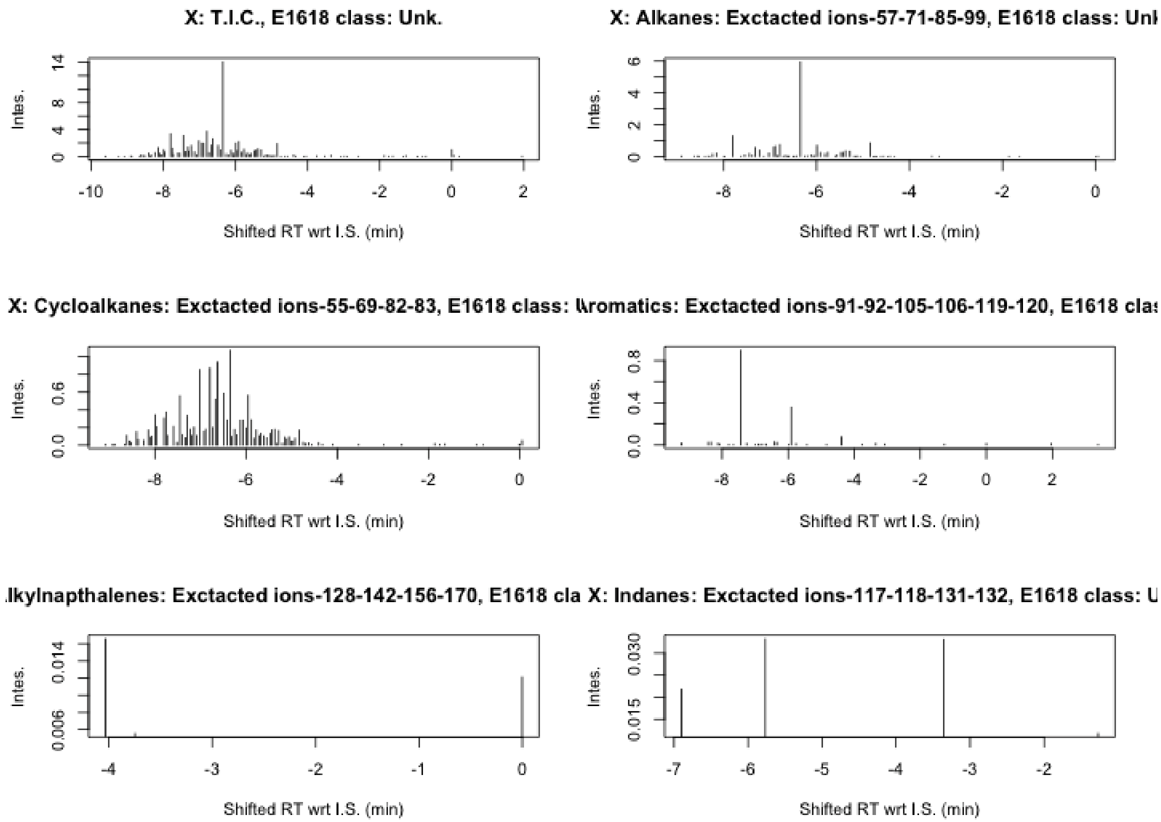
Unknown #	Pred Classes	
4	M Oxy	MPD

Unknown 5: blc59; Crown Lacquer Thinner on Blue Carpet; Med Misc



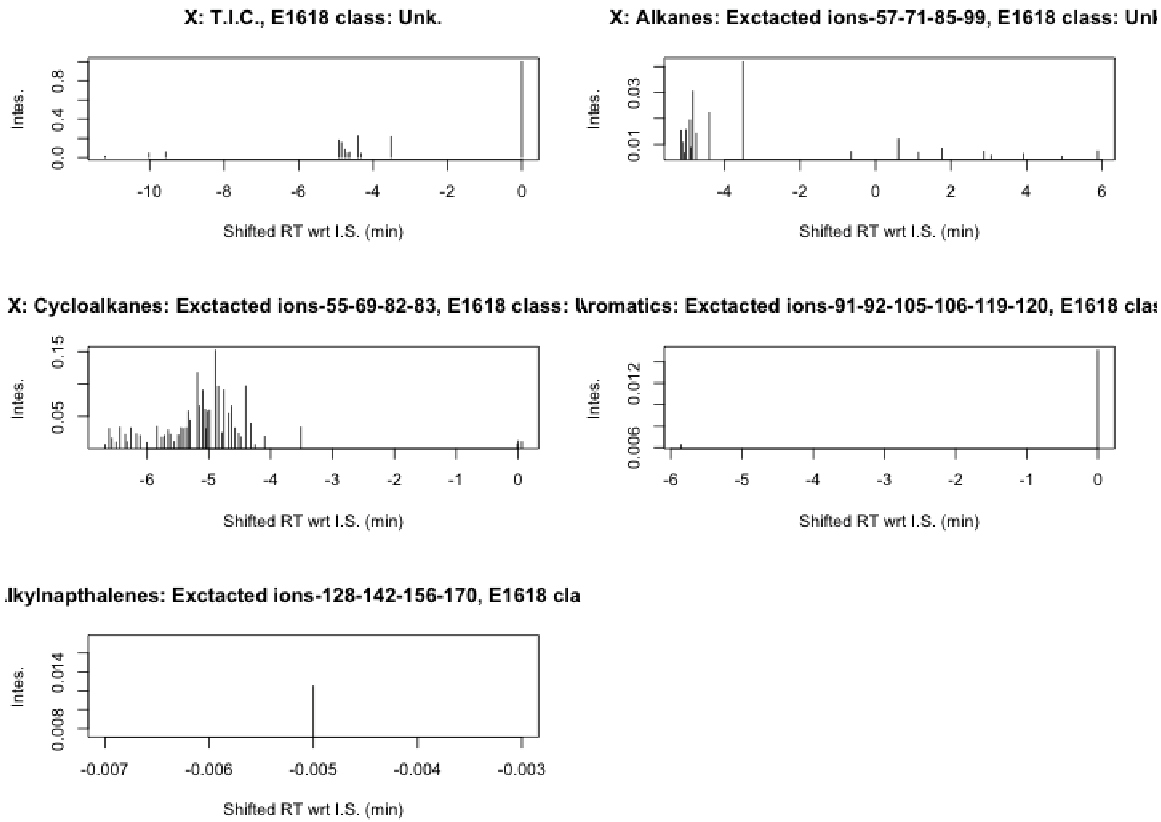
Unknown #	Pred Classes
5	CAN NOT ID

Unknown 6: blc66; Kingsford Charcoal Lighter on Blue Carpet, MDP



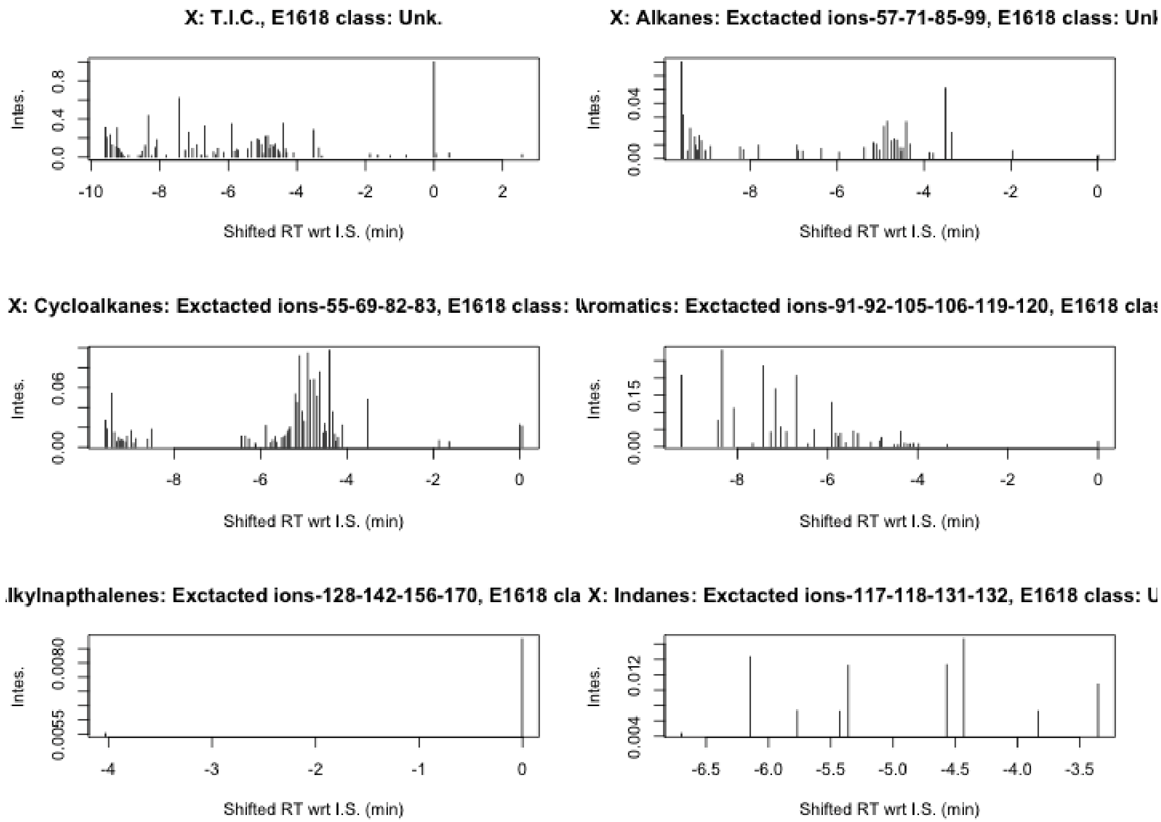
Unknown #	Pred Classes		Human Expert	Notes
6	Med Oxy	MPD		Not picking up possible N-alk pattern at set tol (rt=0.18), but does at rt=0.2.

Unknown 7: blcblk; Blank Sample, Unburned Blue Carpet



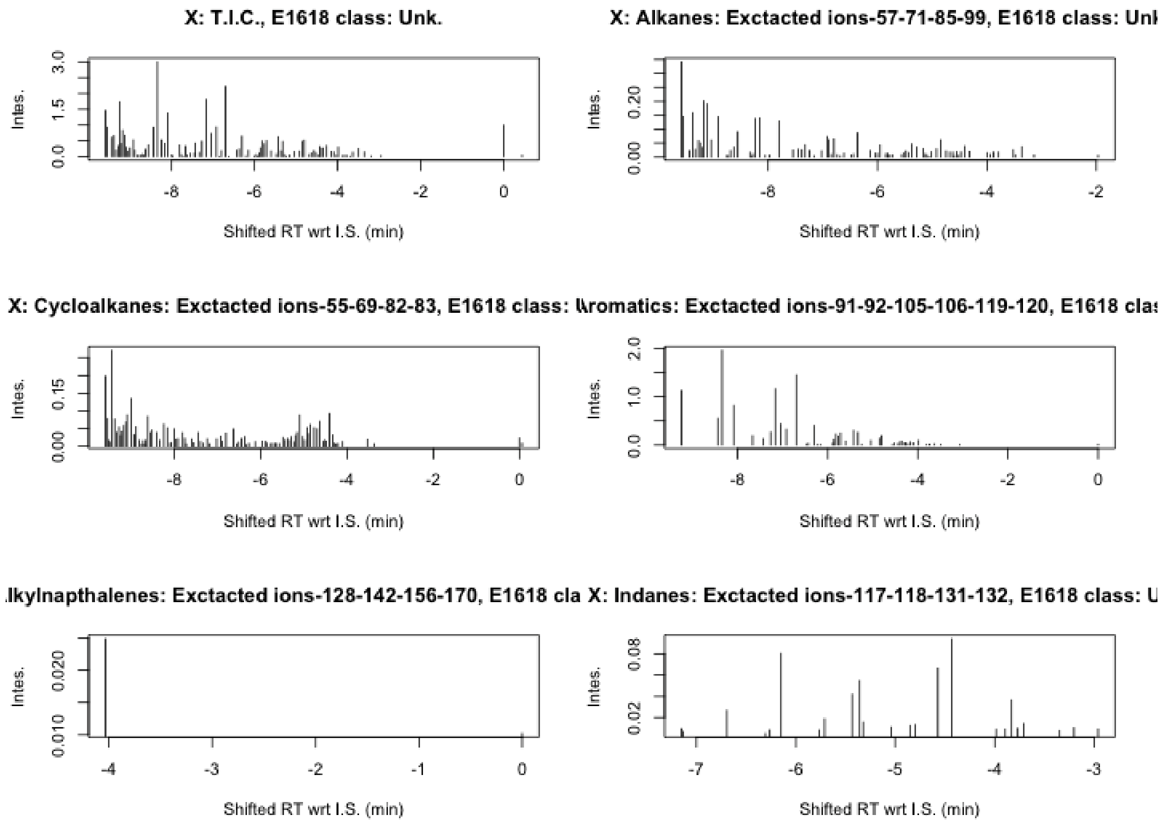
Unknown #	**Pred Classes**
7	CAN NOT ID

Unknown 8: blce85; E85 Gasoline on Blue Carpet



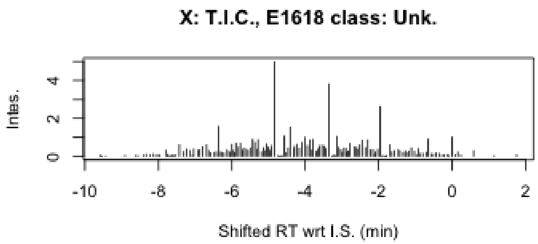
Unknown #	Pred Classes	
8	M Oxy	M Misc

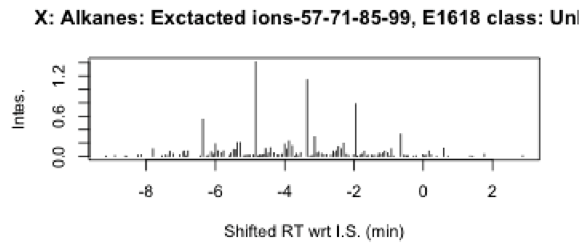
Unknown 9: blcgas; 87 Octane Gasoline on Blue Carpet

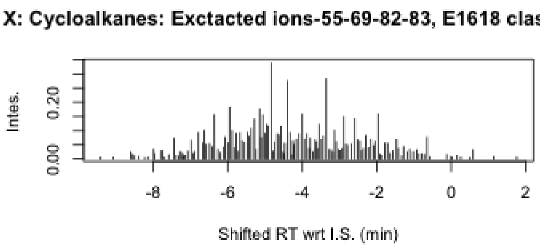


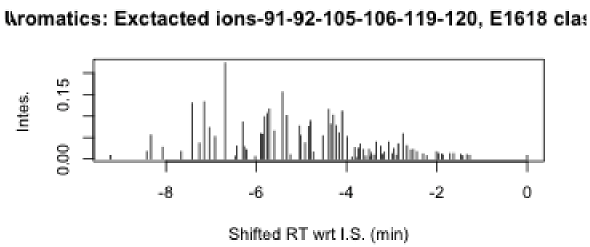
Unknown #	**Pred Classes**
9	M Misc

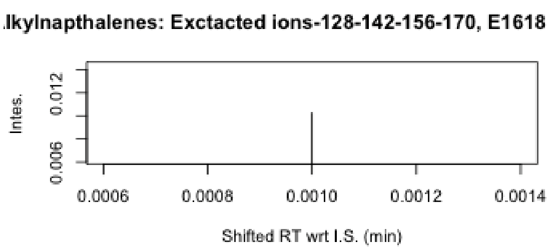
Unknown 10: blcker; Kerosene on Blue Carpet

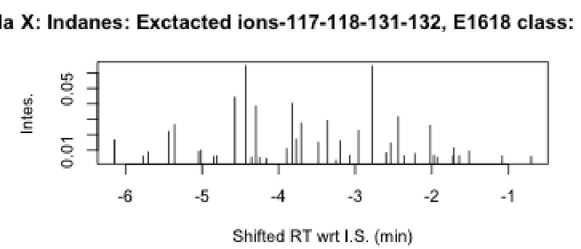






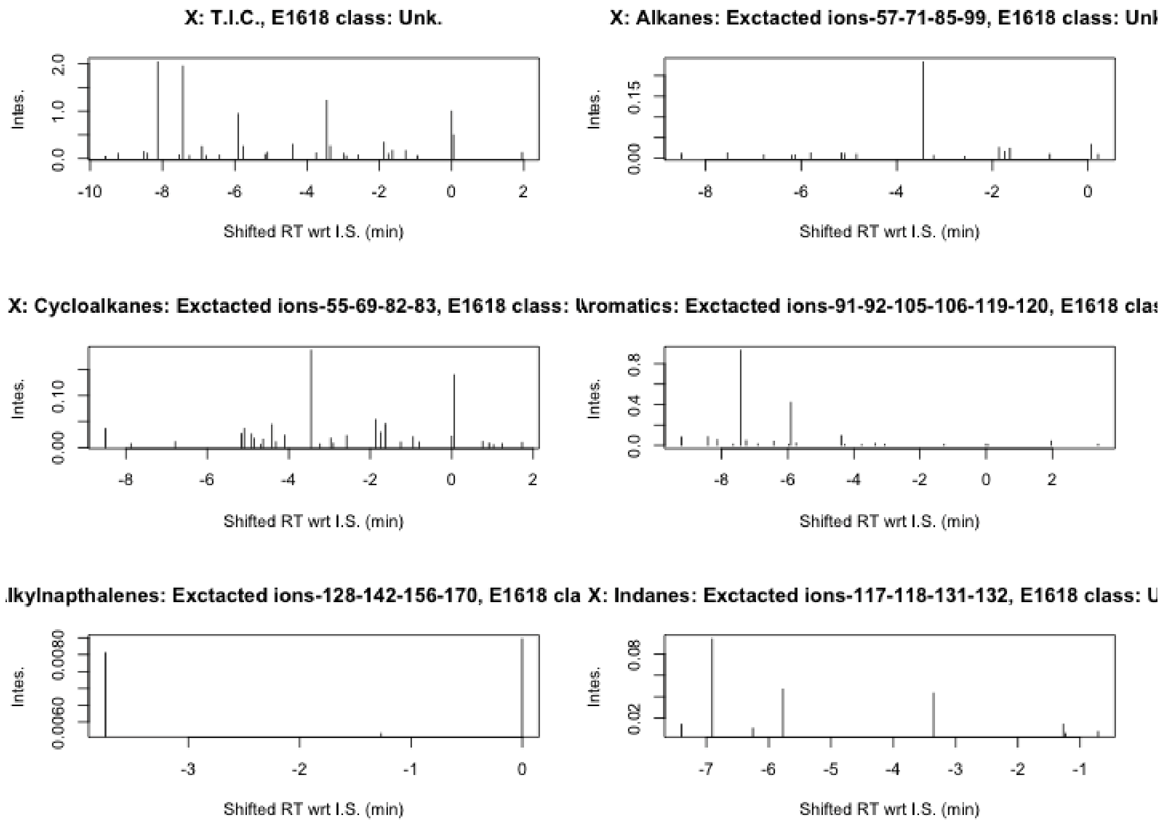






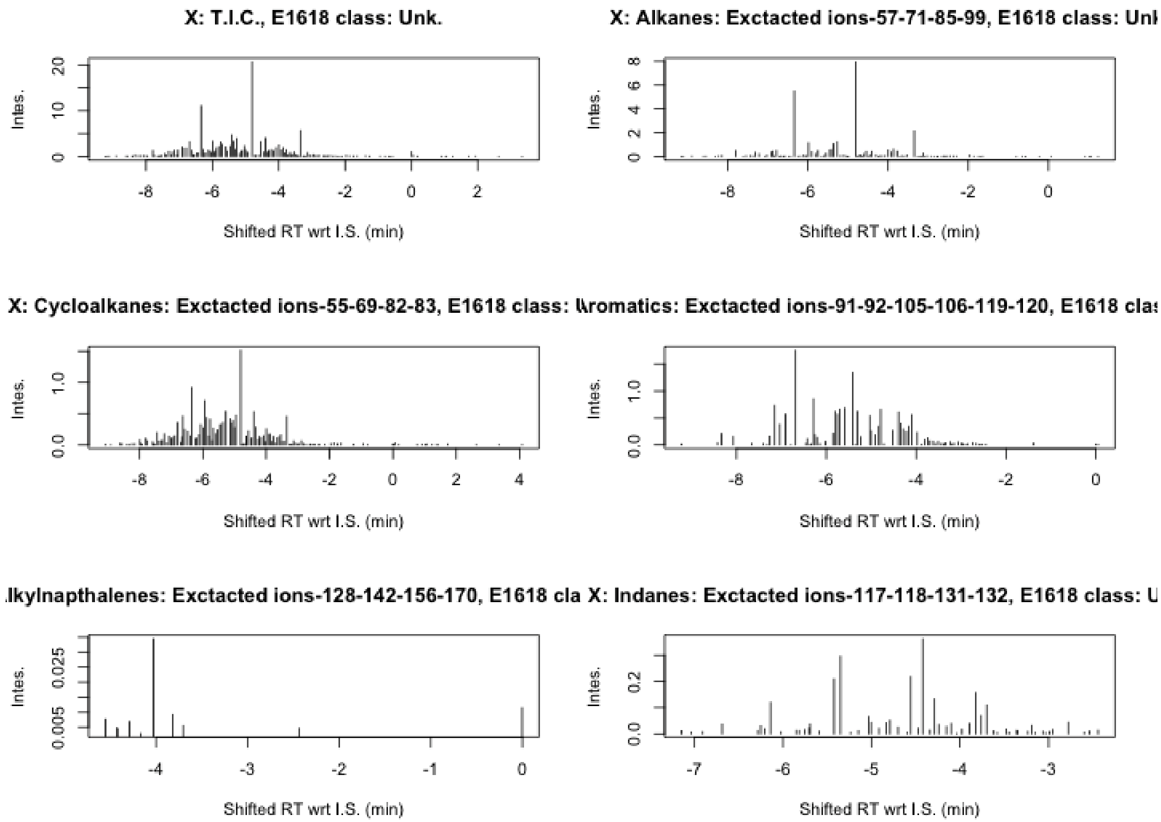
Unknown #	**Pred Classes**		
10	HPD	H Misc	H Oxy

Unknown 11: blcmtdb; Burned Blank Blue Carpet



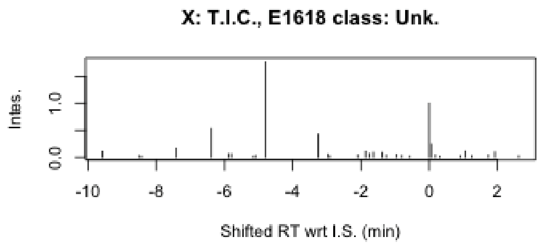
Unknown #	Pred Classes	
11	H Misc	HPD

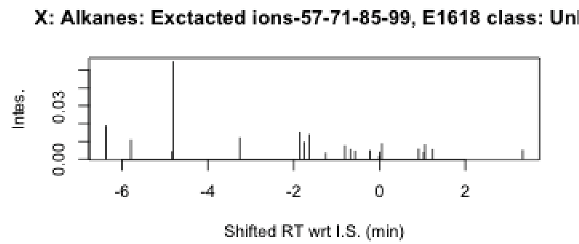
Unknown 12: brc54; Crownh Paint Thinner on Brown Carpet; MPD

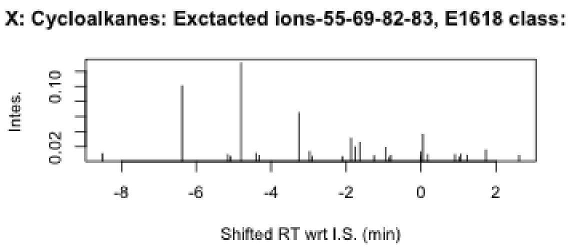


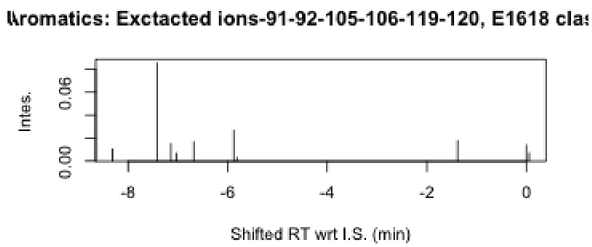
Unknown #	Pred Classes	
12	MPD	M Oxy

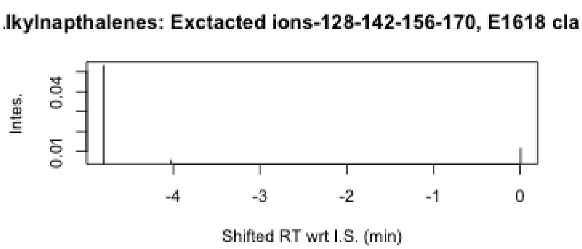
Unknown 13: brc59; Crown Lacquer Thinner on Brown carpet; Med Misc





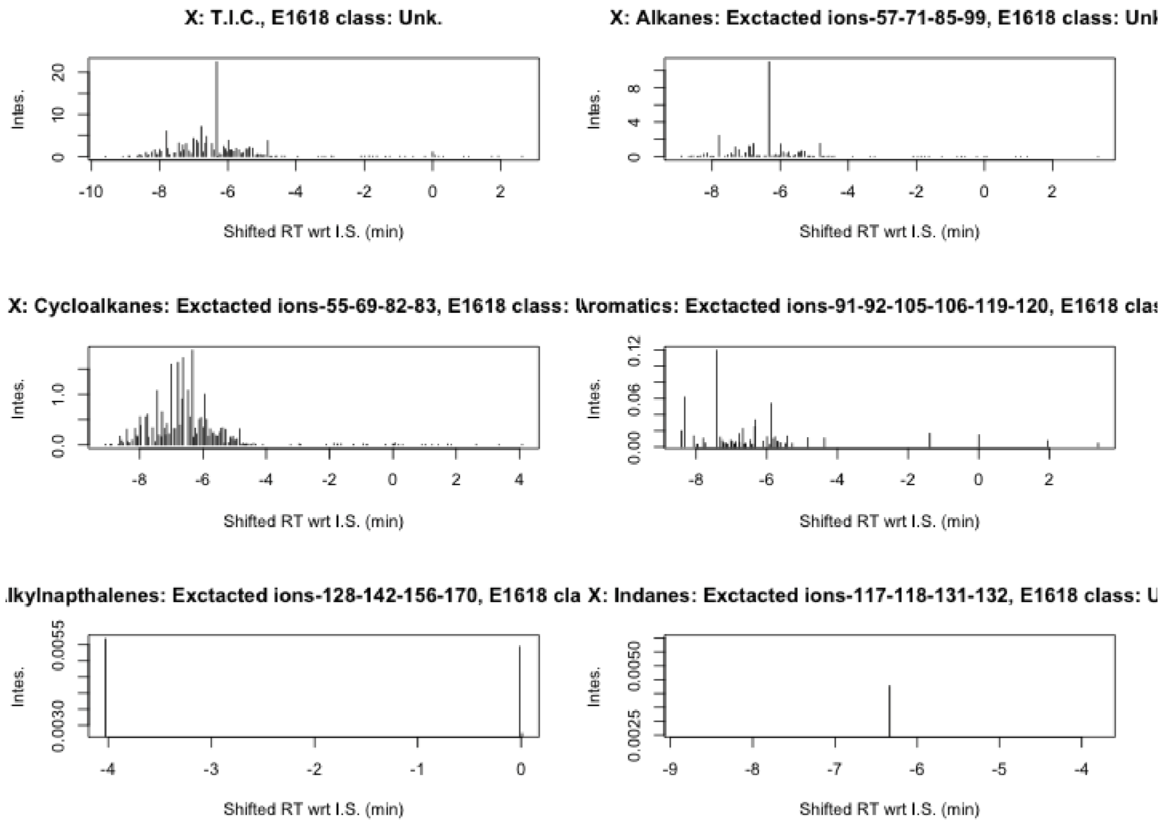






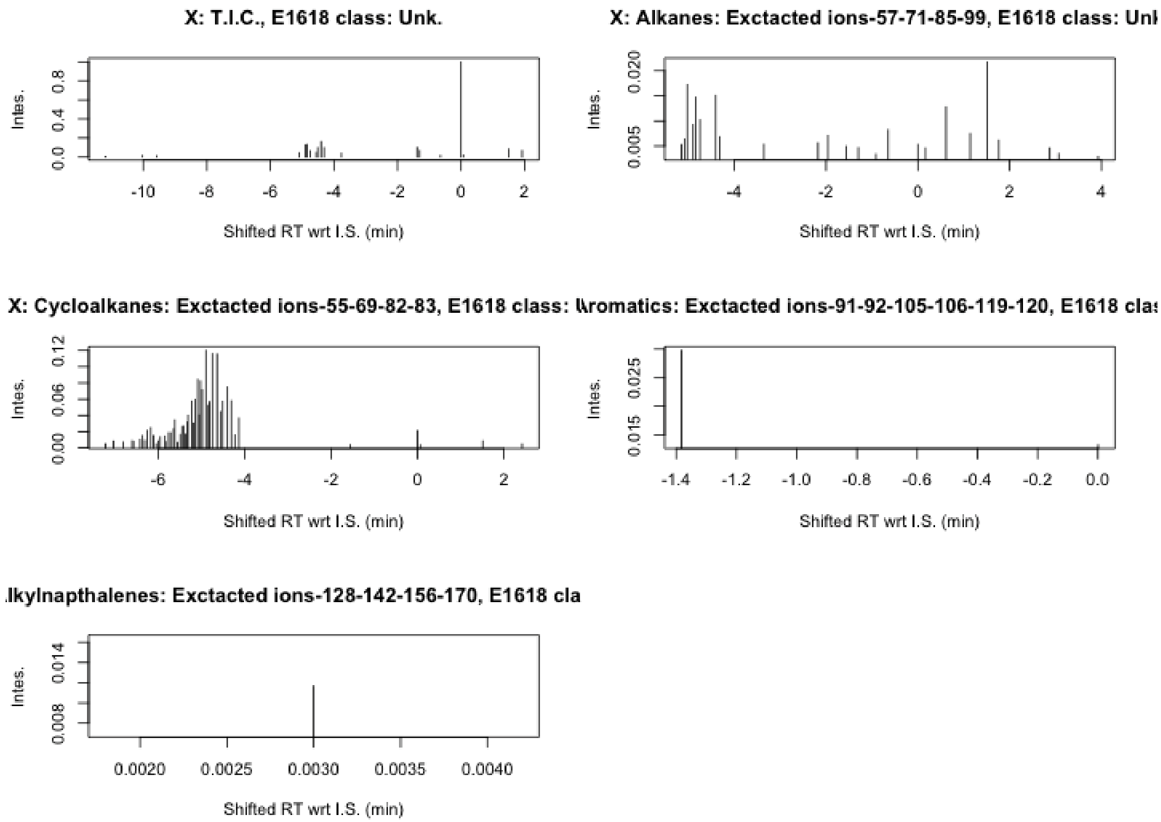
Unknown #	Pred Classes
13	H Oxy
	CAN NOT ID

Unknown 14: brc66; Kingsford Charcoal Lighter on Brown Carpet; MPD



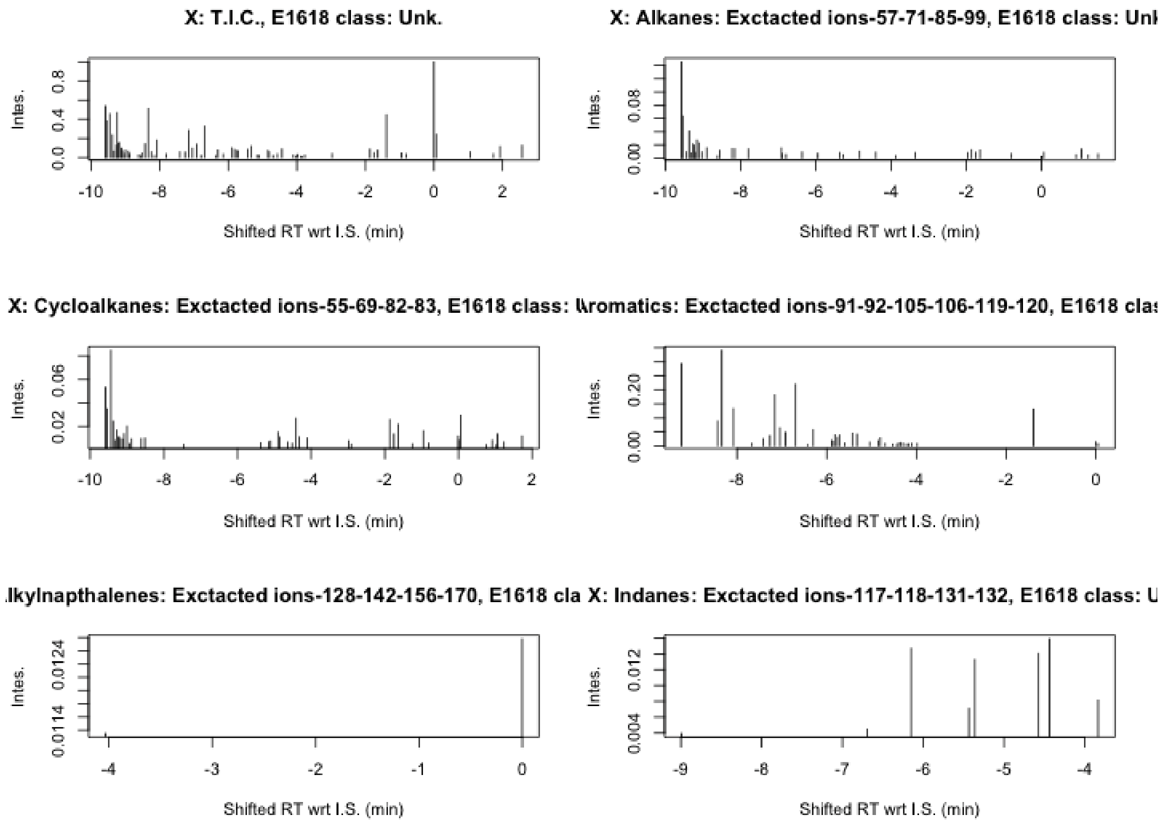
Unknown #	Pred Classes	
14	M Oxy	MPD

Unknown 15: brcblk; Blank, Unburned Brown Carpet



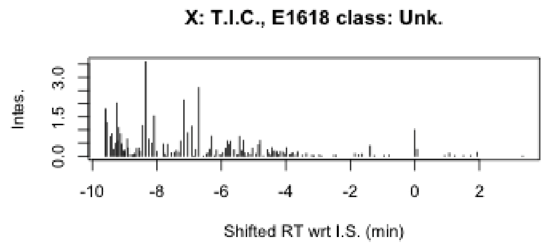
Unknown #	**Pred Classes**
15	CAN NOT ID

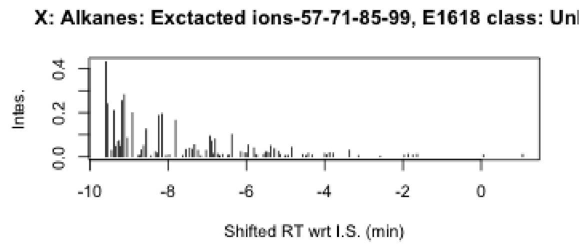
Unknown 16: brce85; E85 Gasoline on Brown Carpet

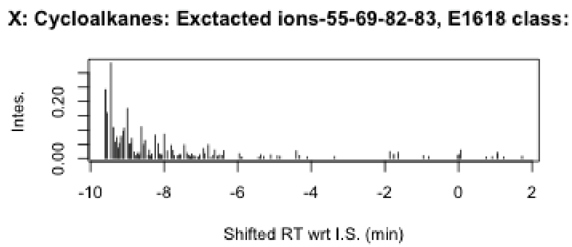


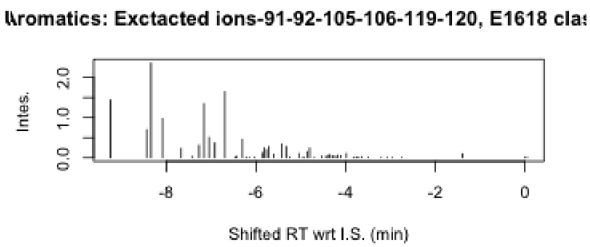
Unknown #	Pred Classes
16	M Misc

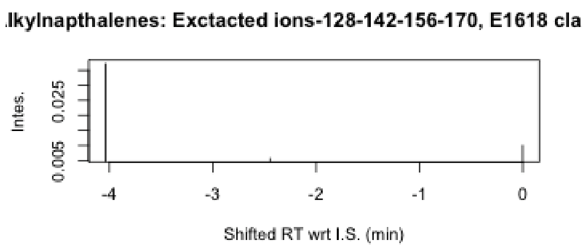
Unknown 17: brcgas; Gasoline on Brown Carpet' Gas

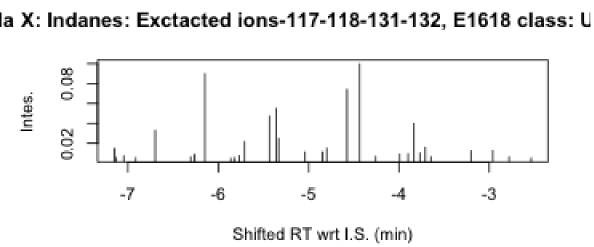






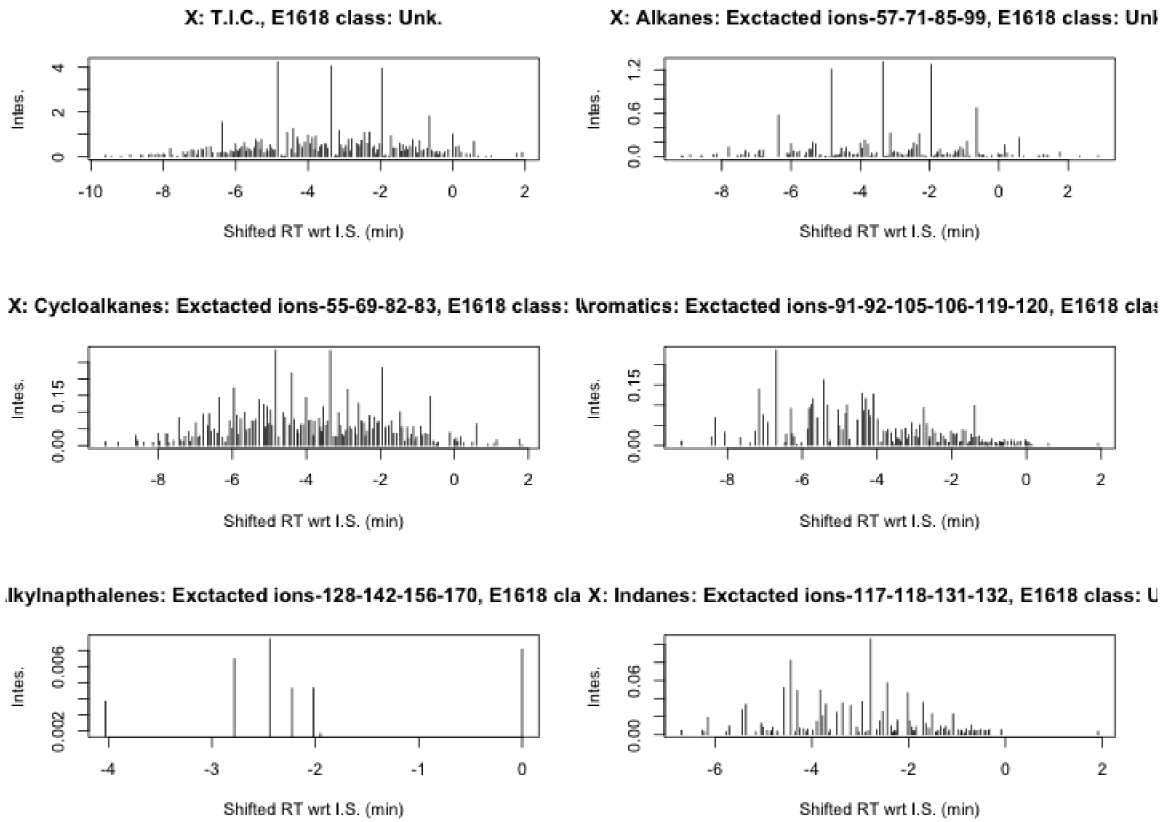






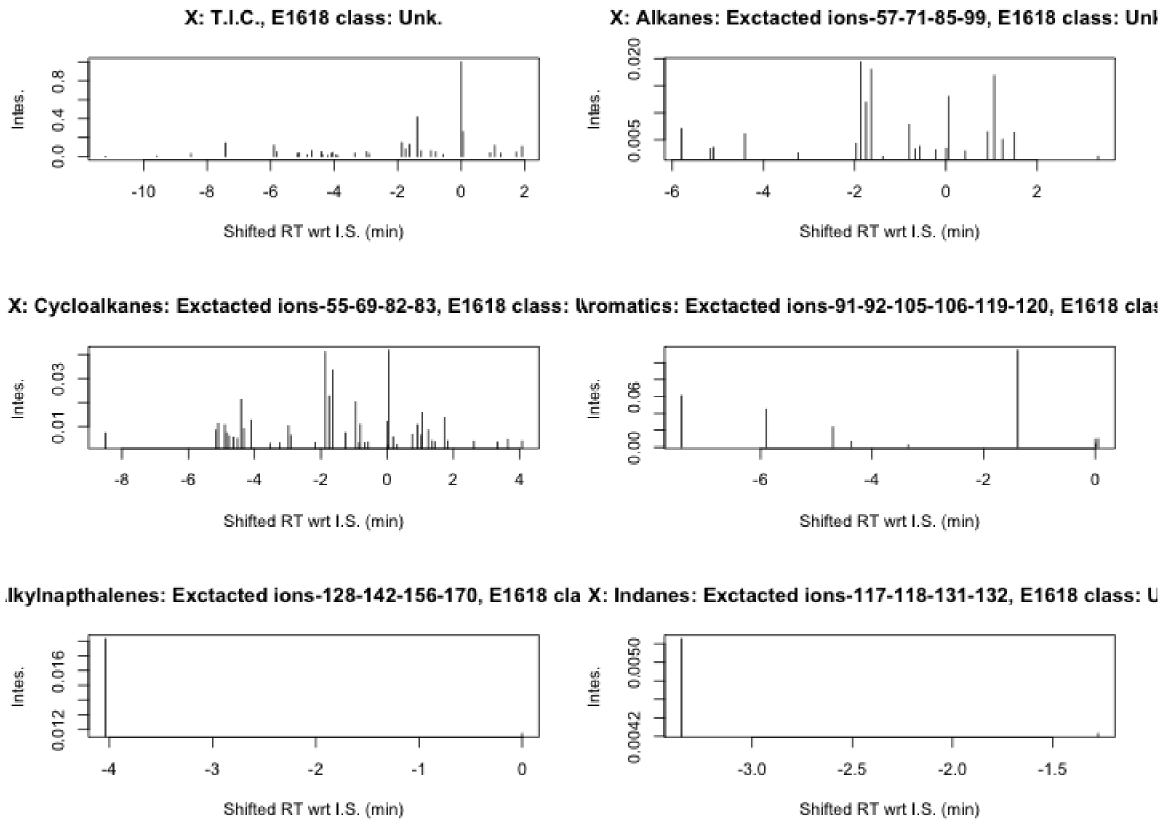
Unknown #	Pred Classes	
17	Med Misc	
	Med Misc	MPD

Unknown 18: brcker; Kerosene on Brown Carpet



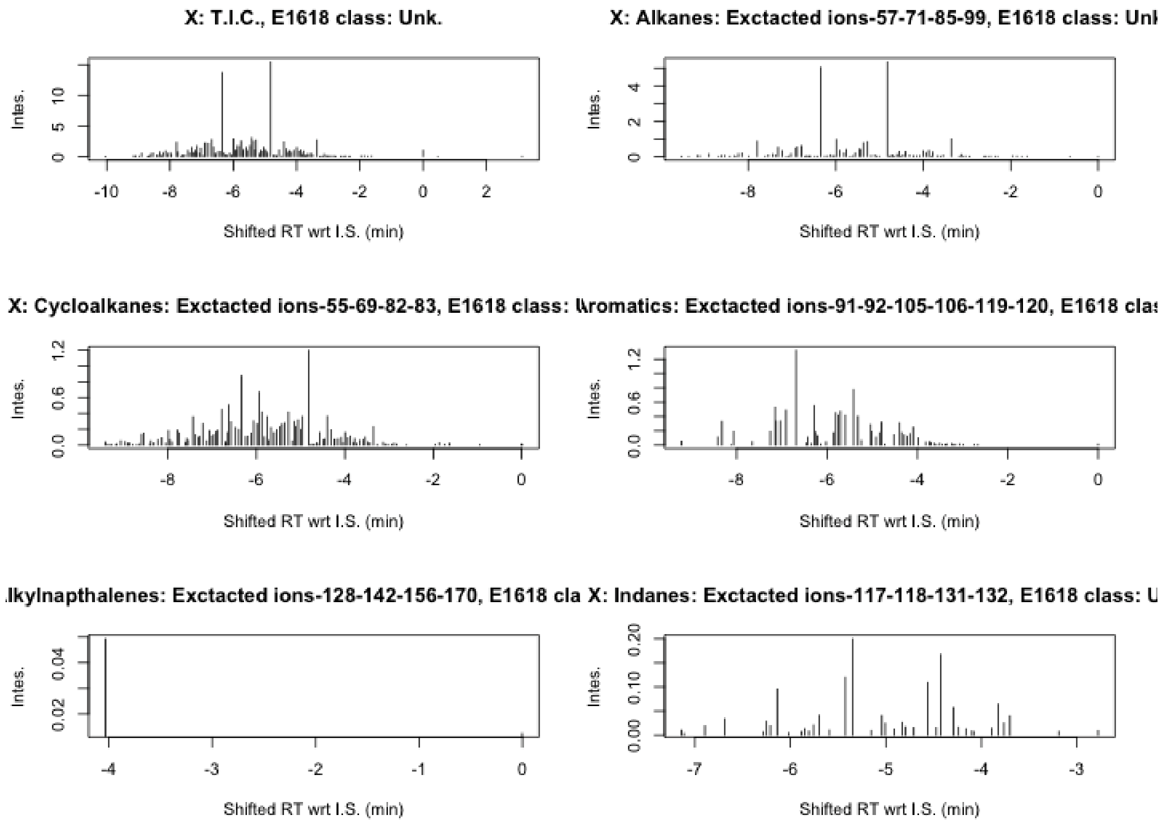
Unknown #	Pred Classes		
18	HPD	H Misc	H Oxy

Unknown 19: brcmtdb; Blank Burned Brown Carpet



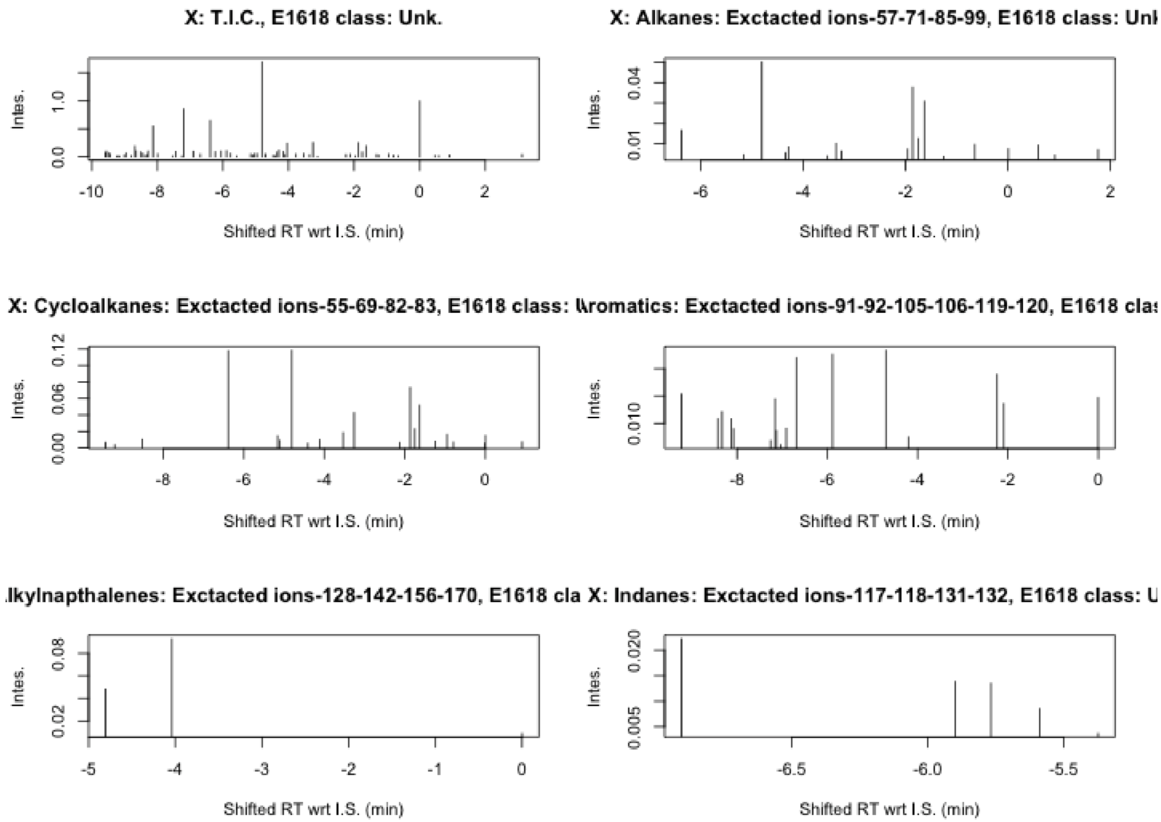
Unknown #	**Pred Classes**
19	CAN NOT ID

Unknown 20: cp54; Crown Paint Thinner on Carpet Pad, MPD



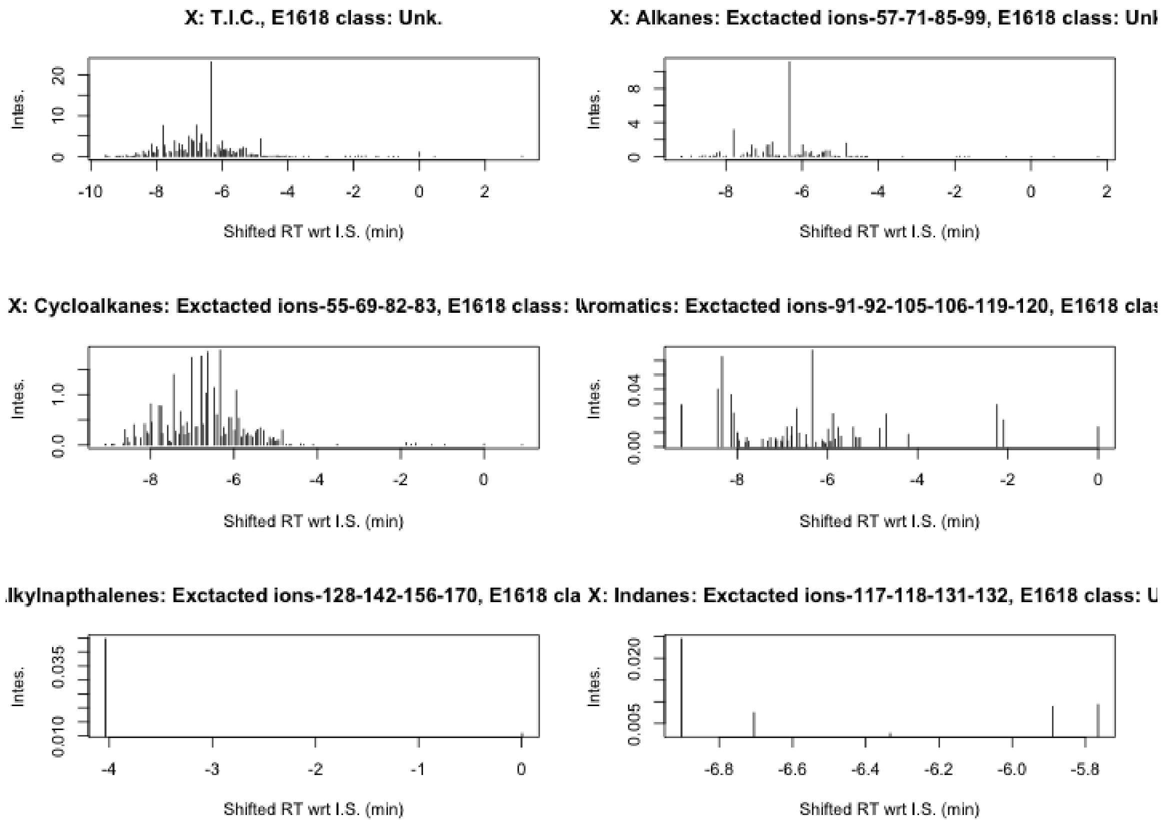
Unknown #	**Pred Classes**		
20	M Nap-Par	M Oxy	
	MPD	M Oxy	M Misc

Unknown 21: cp59; Crown Lacquer Thinner on Carpet Pad; Med Misc



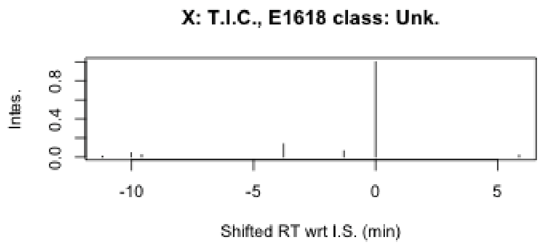
Unknown #	Pred Classes
21	CAN NOT ID
	CAN NOT ID

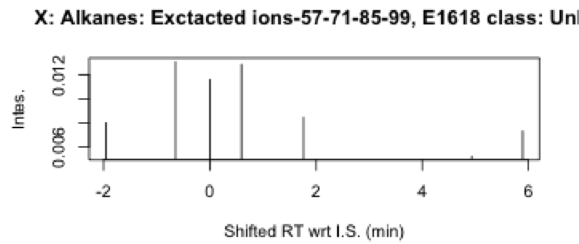
Unknown 22: cp66; Kingsford Charcoal Lighter on Carpet Pad; MPD

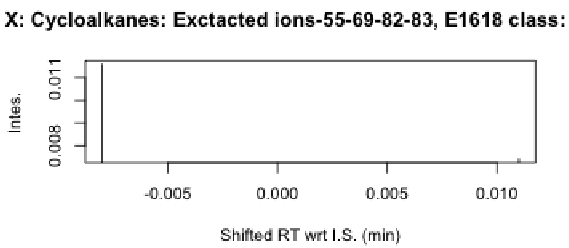


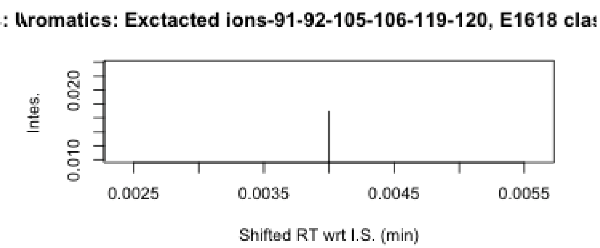
Unknown #	**Pred Classes**	
22	M Oxy	MPD
	MPD	M Misc

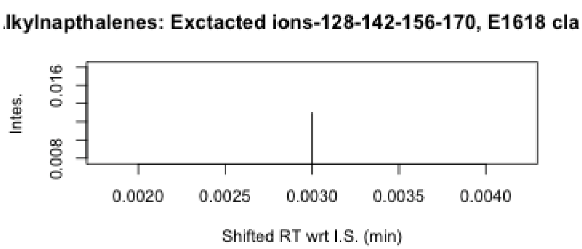
Unknown 23: cpblk; Blank unburned carpet pad





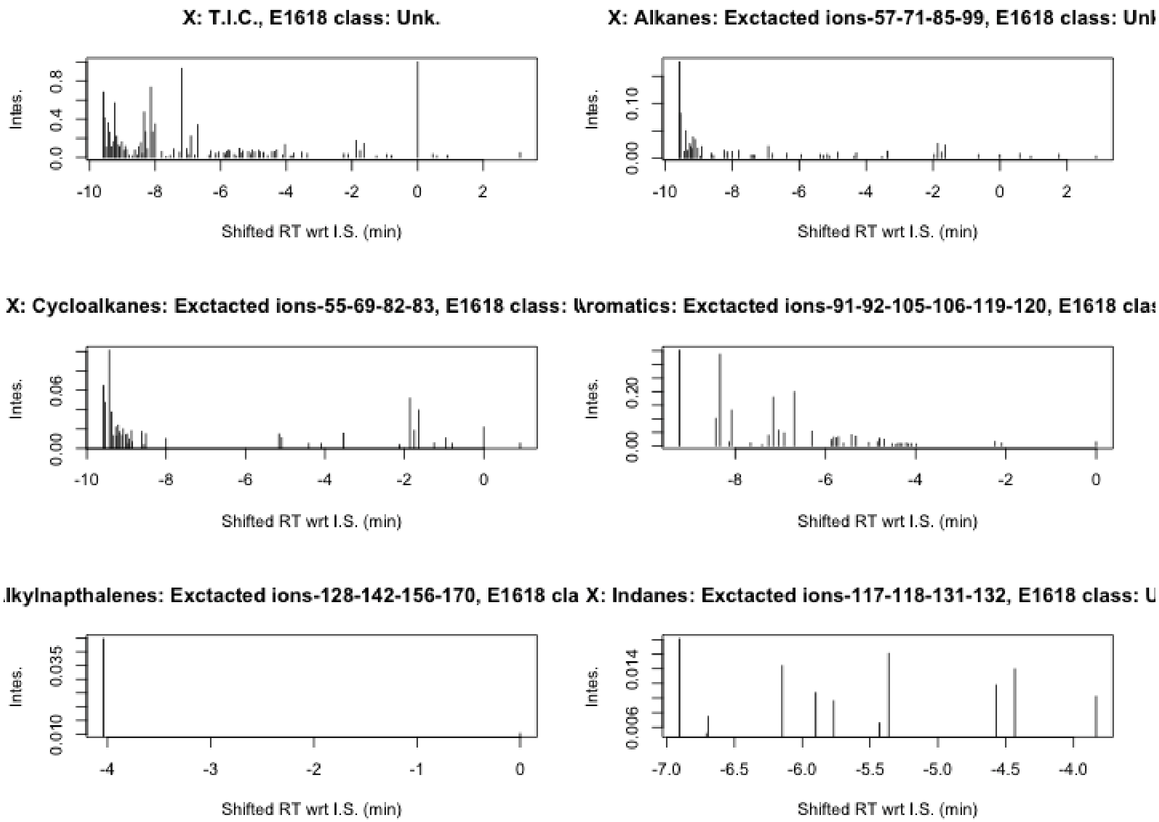






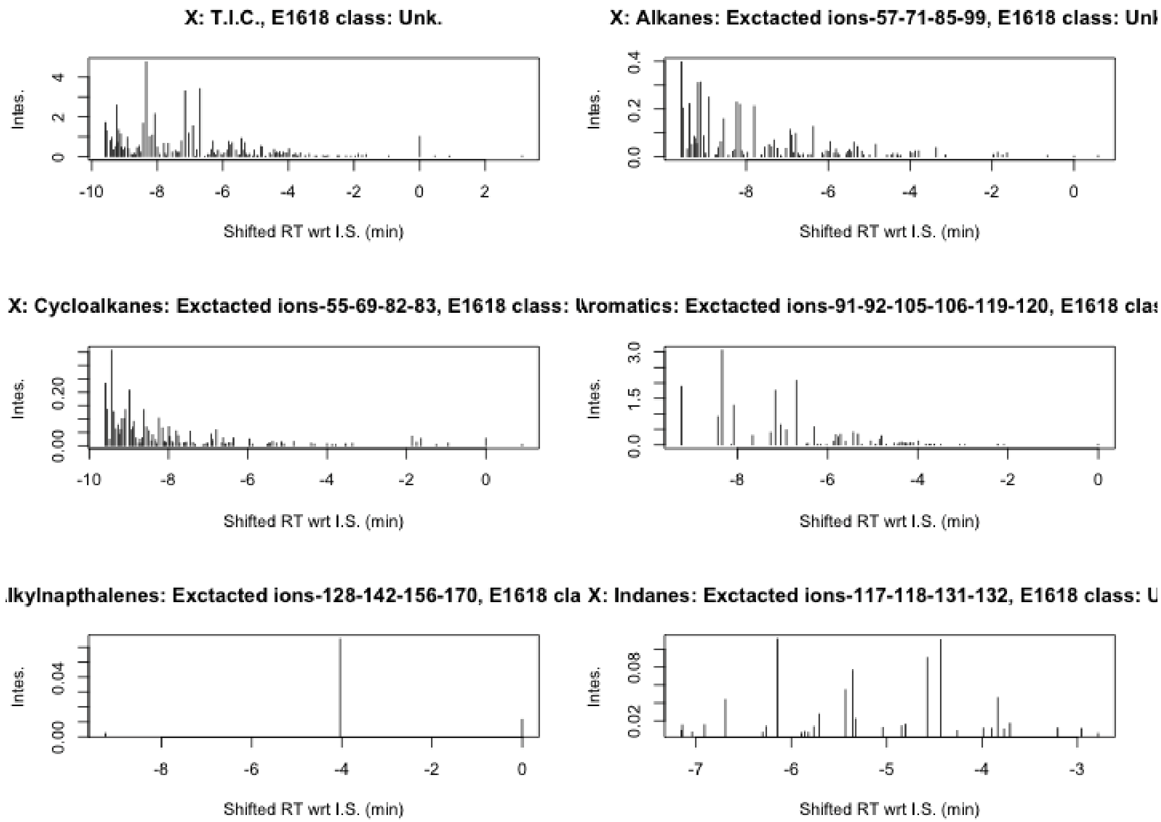
Unknown #	Pred Classes
23	CAN NOT ID

Unknown 24: cpe85; E85 gasoline on carpet pad



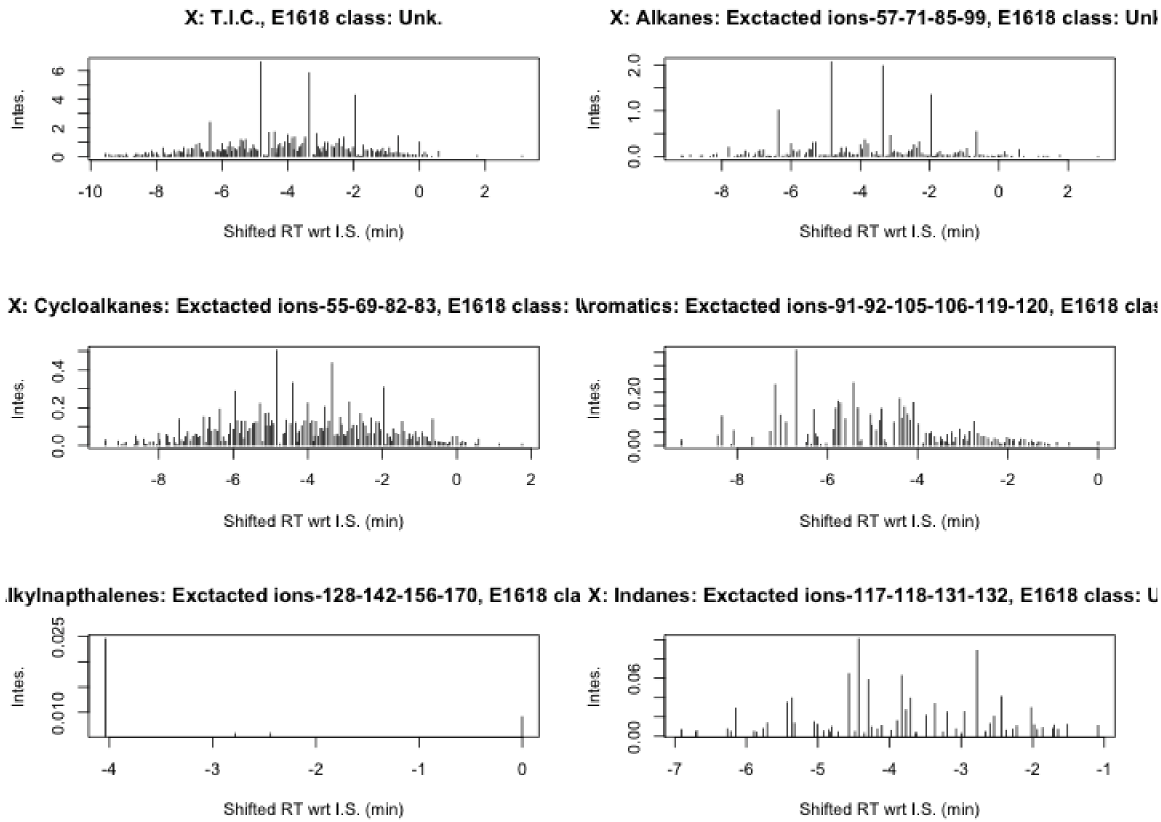
Unknown #	Pred Classes
24	Med Oxy

Unknown 25: cpgas; Gasoline on Carpet pad; GAS



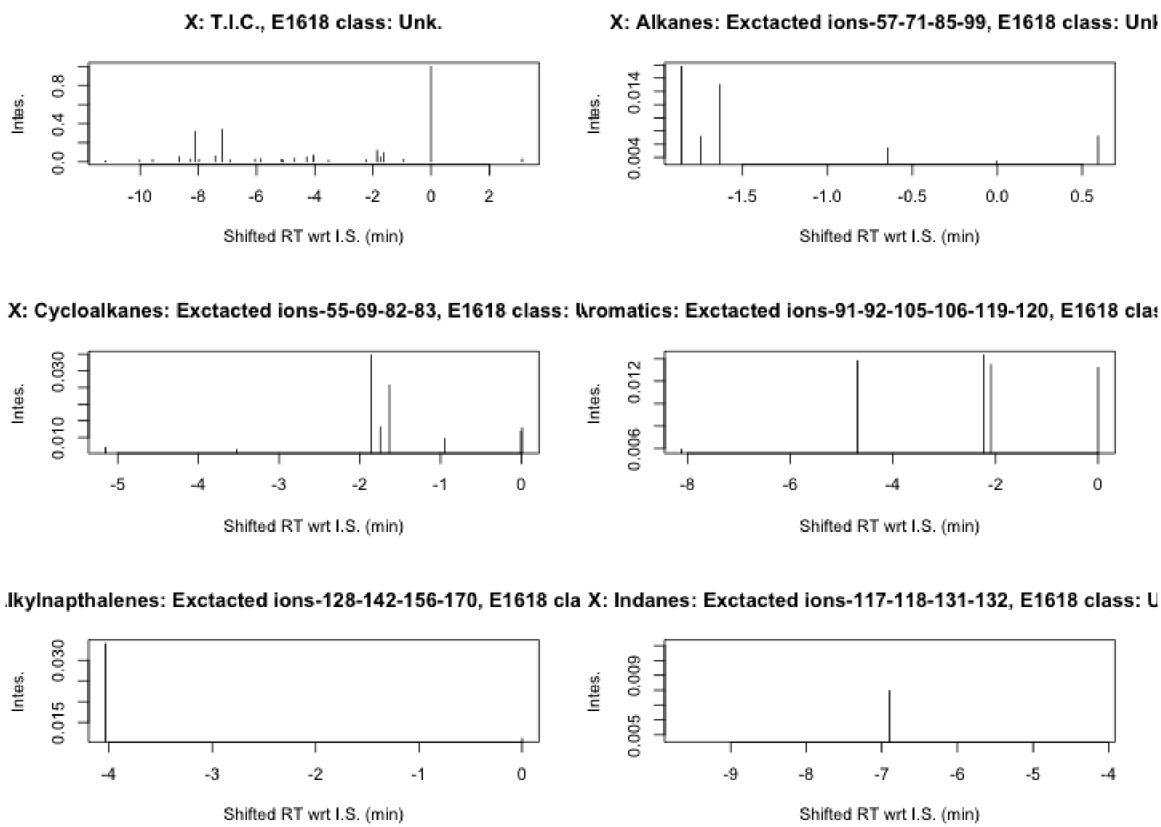
Unknown #	**Pred Classes**
25	Med Misc

Unknown 26: cpker; Kerosene on Carpet Pad; HPD



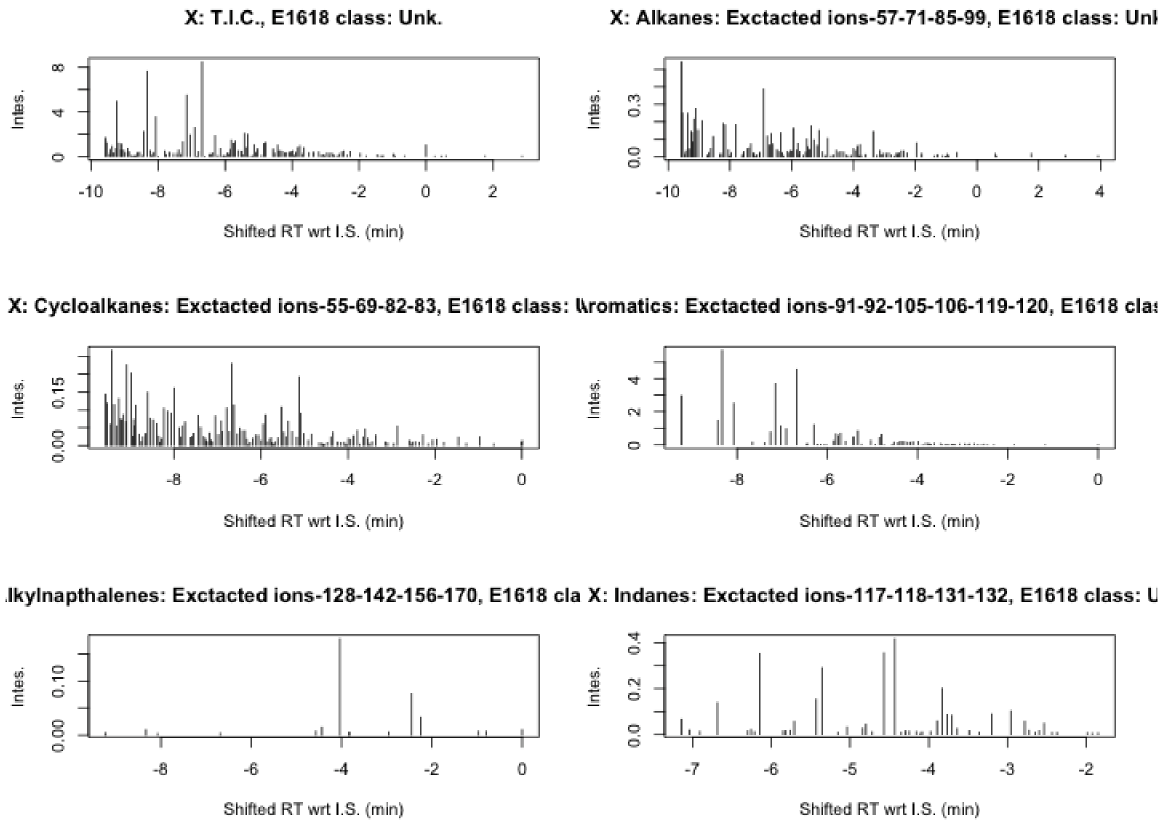
Unknown #	Pred Classes		
26	HPD	H Misc	H Oxy

Unknown 27: cpmtdb; Blank burnt carpet pad



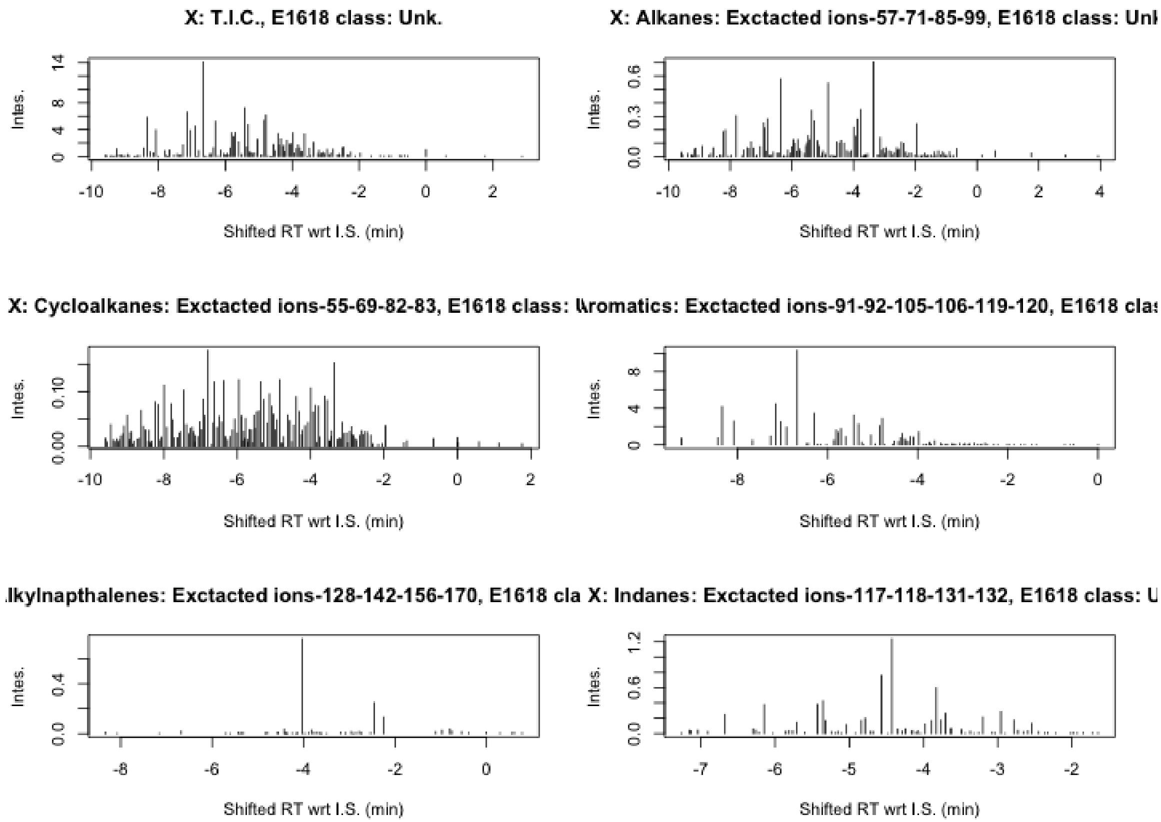
Unknown #	Pred Classes
27	CAN NOT ID

Unknown 28: e85yp; E85 gasoline on Yellow Pine; GAS



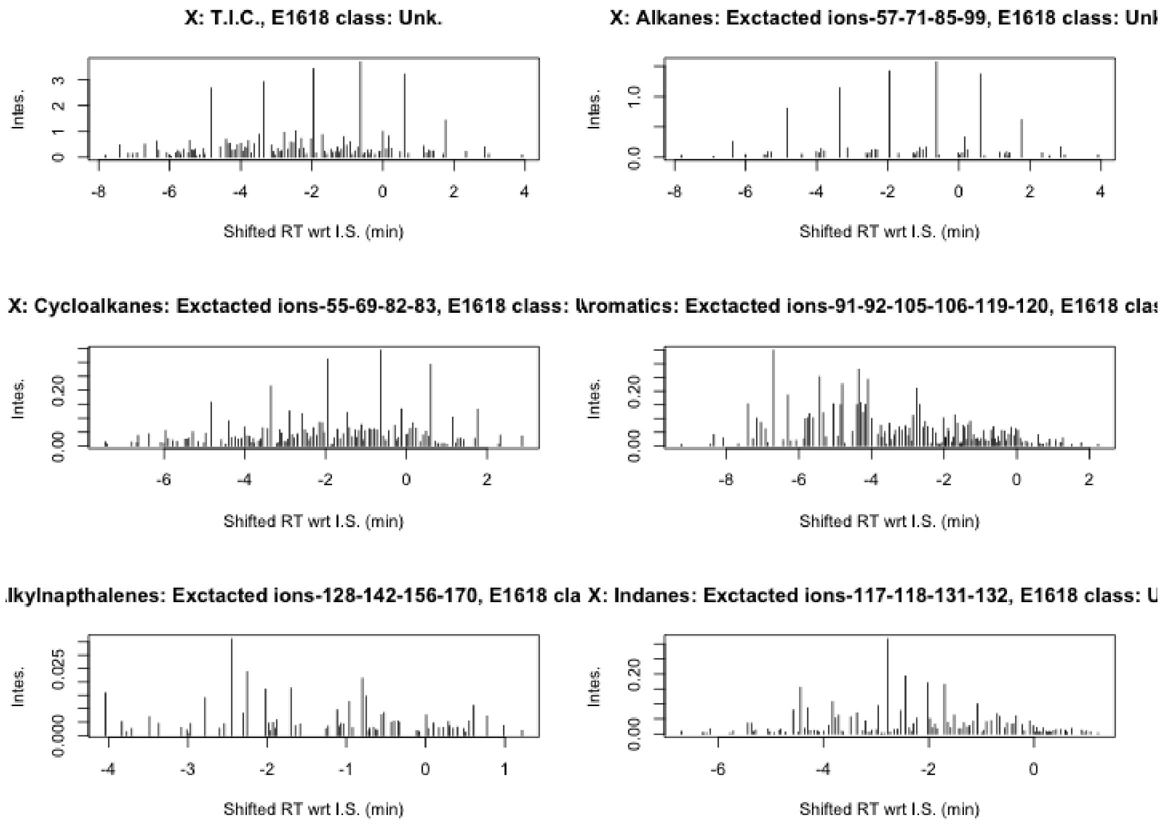
Unknown #	Pred Classes	
28	M Misc	
	M Misc	MPD

Unknown 29: gasyp; Gasoline on Yellow Pine; GAS



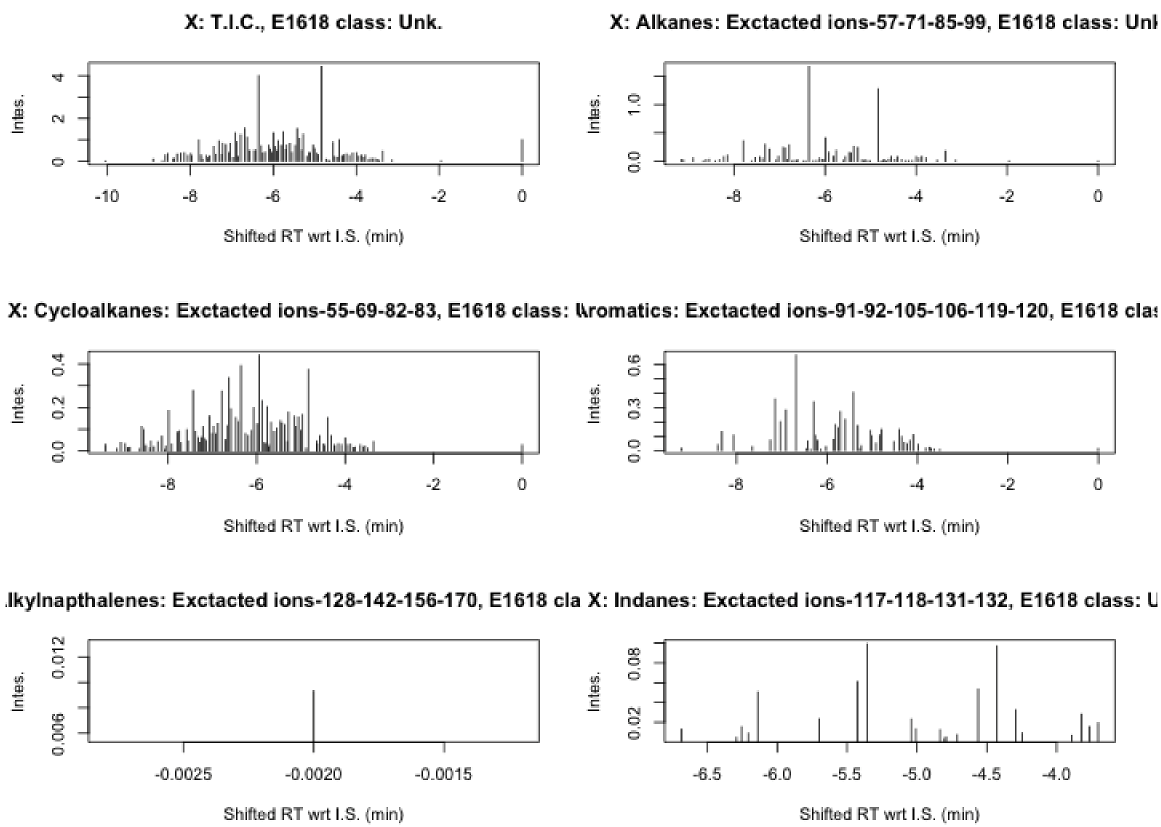
Unknown #	**Pred Classes**
29	M Misc

Unknown 30: keryp; Kerosene on Yellow Pine



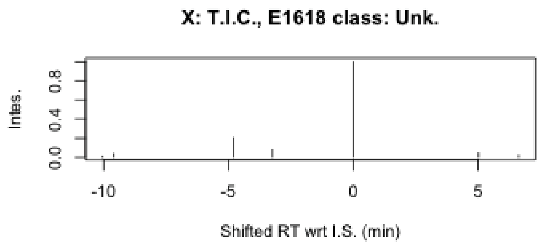
Unknown #	Pred Classes
30	CAN NOT ID

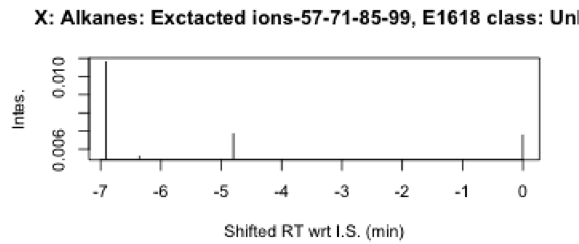
Unknown 31: oo54;

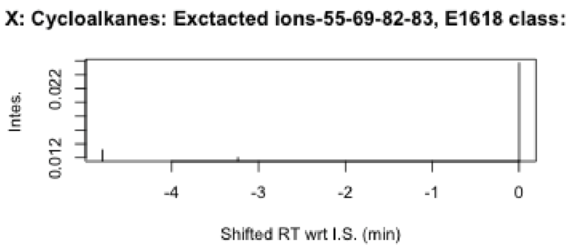


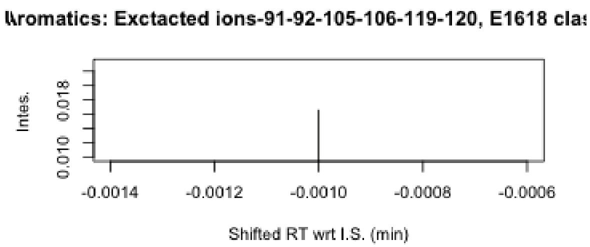
Unknown #	Pred Classes		
31	M Nap-par	M Oxy	
	MPD	M Misc	M Oxy

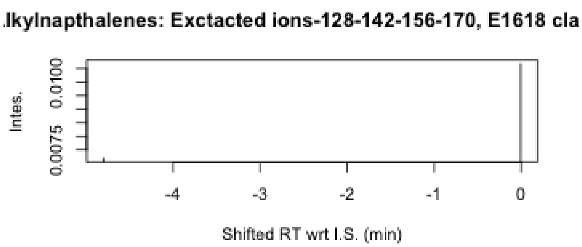
Unknown 32: oo59





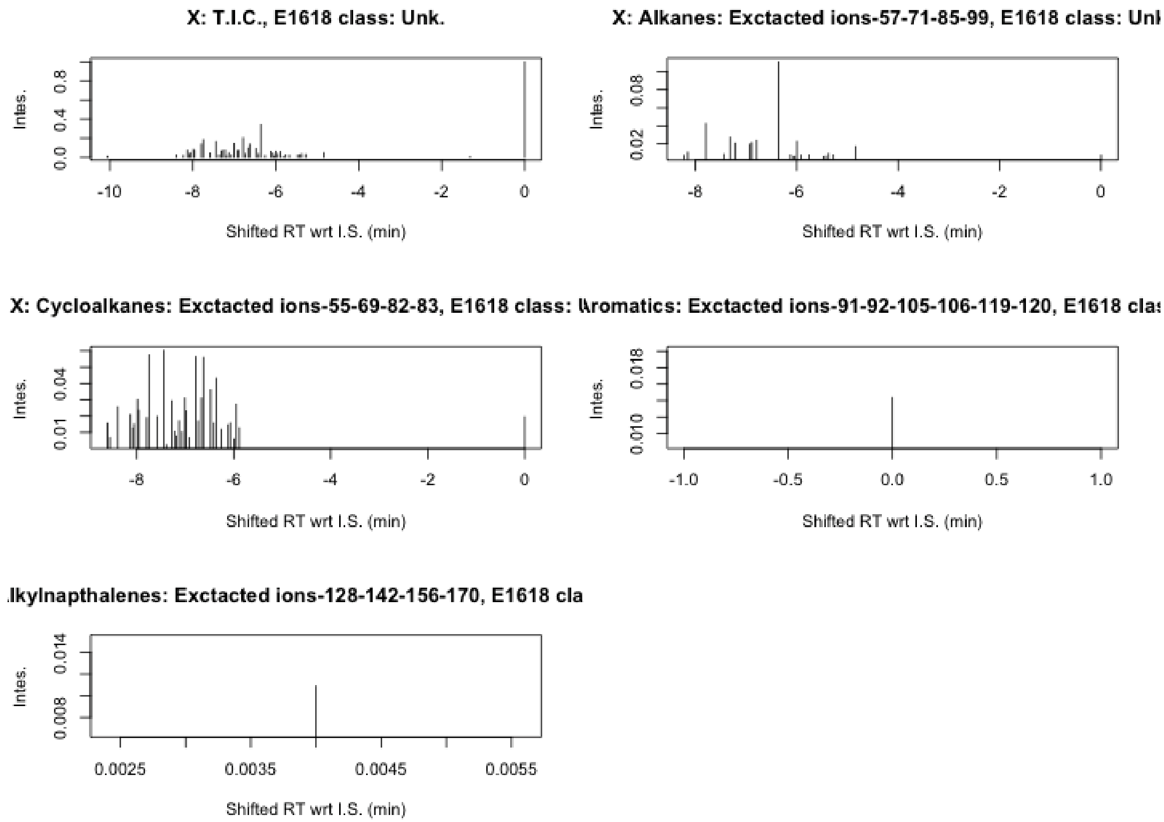






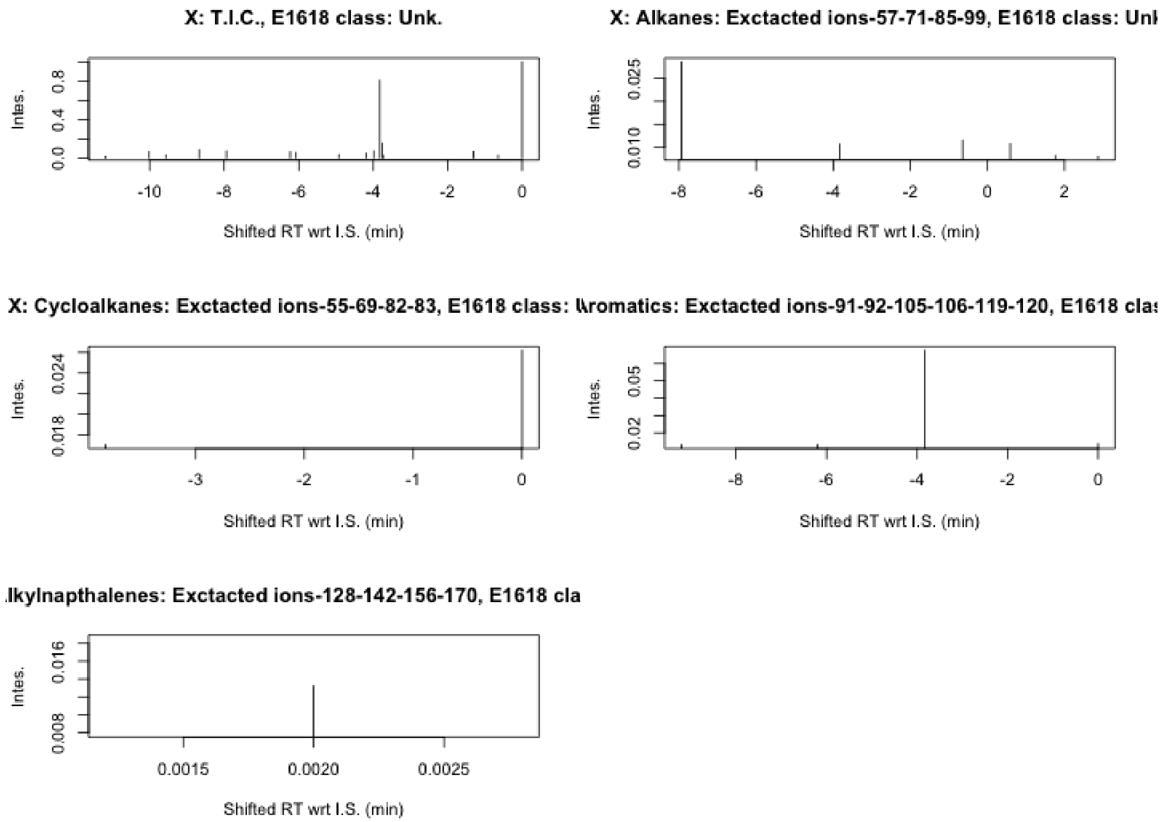
Unknown #	**Pred Classes**
32	H Oxy

Unknown 33: oo66



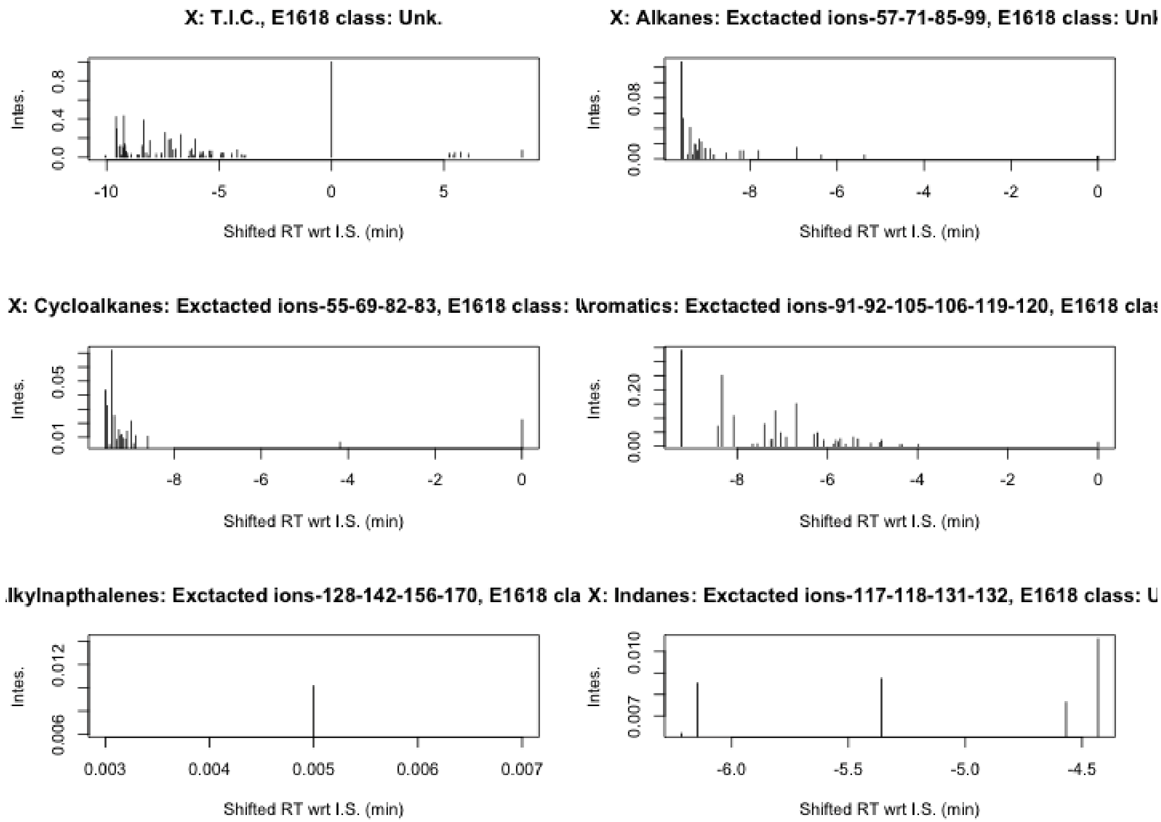
Unknown #	Pred Classes
33	MPD

Unknown 34: ooblk



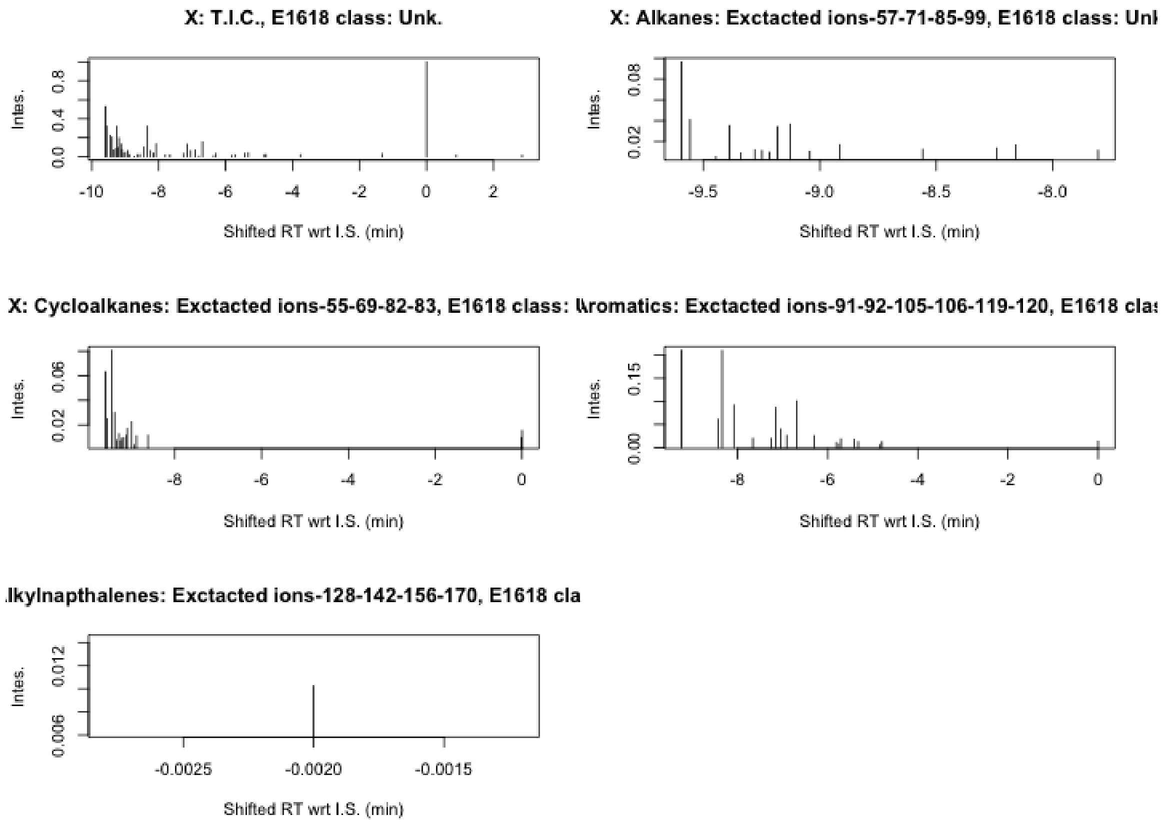
Unknown #	Pred Classes
34	CAN NOT ID

Unknown 35: ooe85



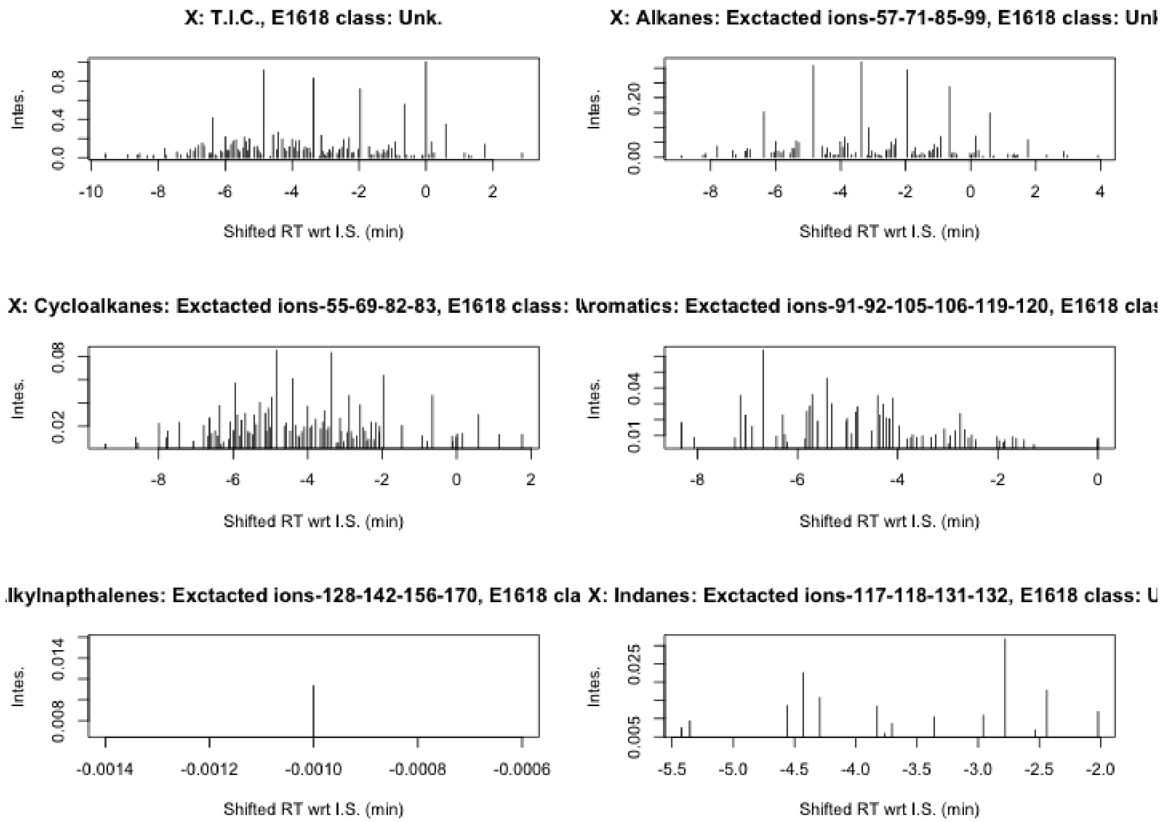
Unknown #	Pred Classes
35	M Misc

Unknown 36: oogas



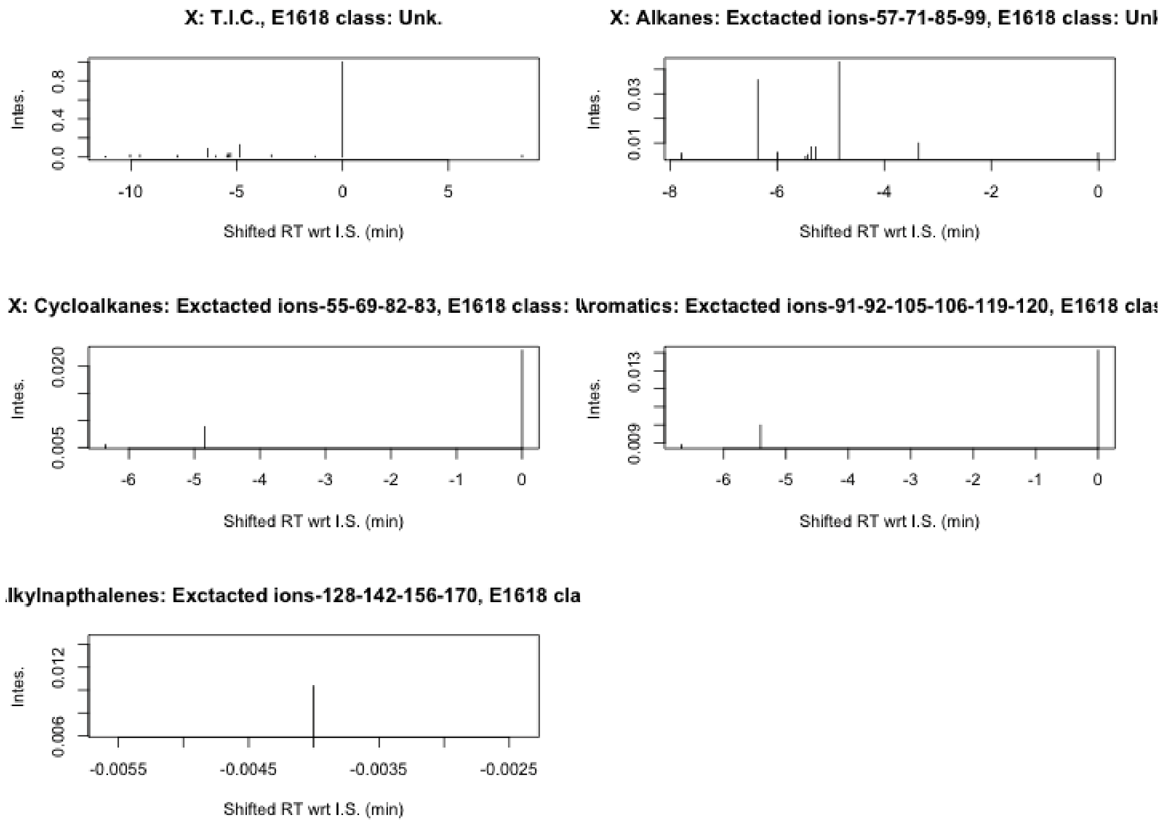
Unknown #	Pred Classes		
36	M Misc	M Oxy	MPD

Unknown 37: ooker



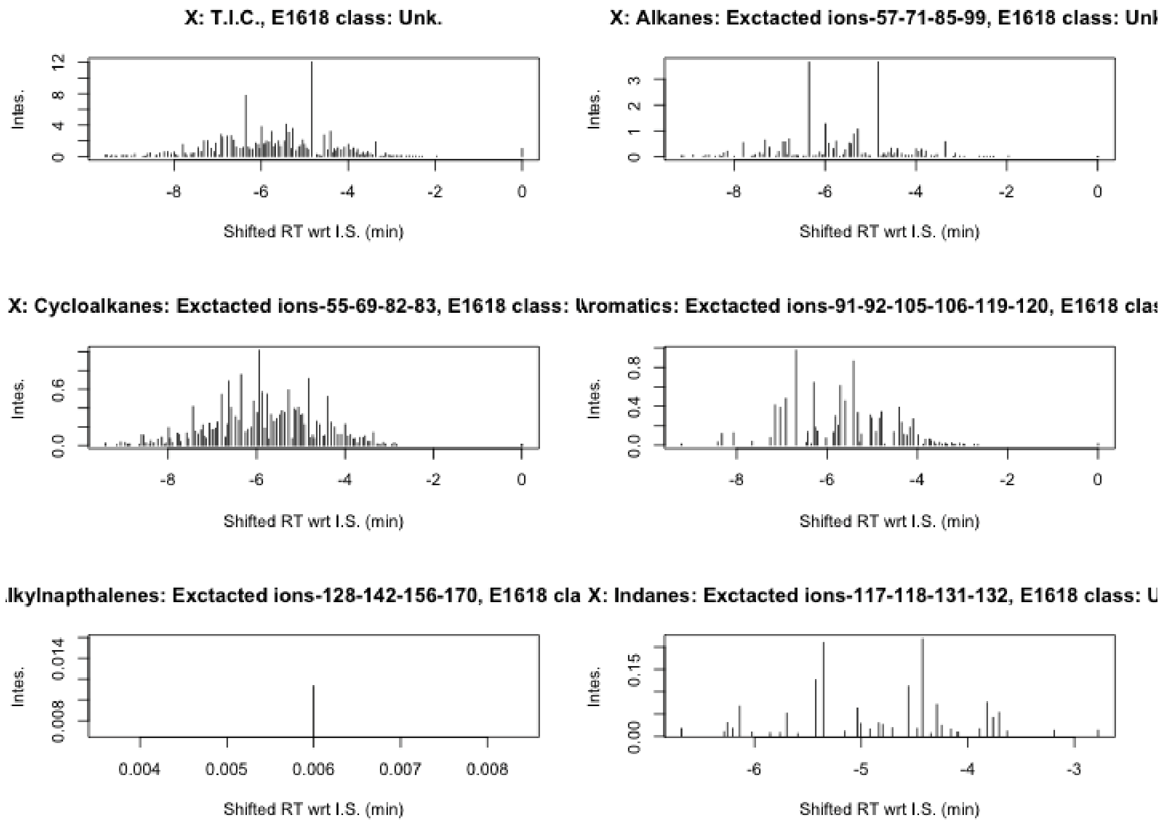
Unknown #	Pred Classes		
37	HPD	H Misc	H Oxy

Unknown 38: oomtdb



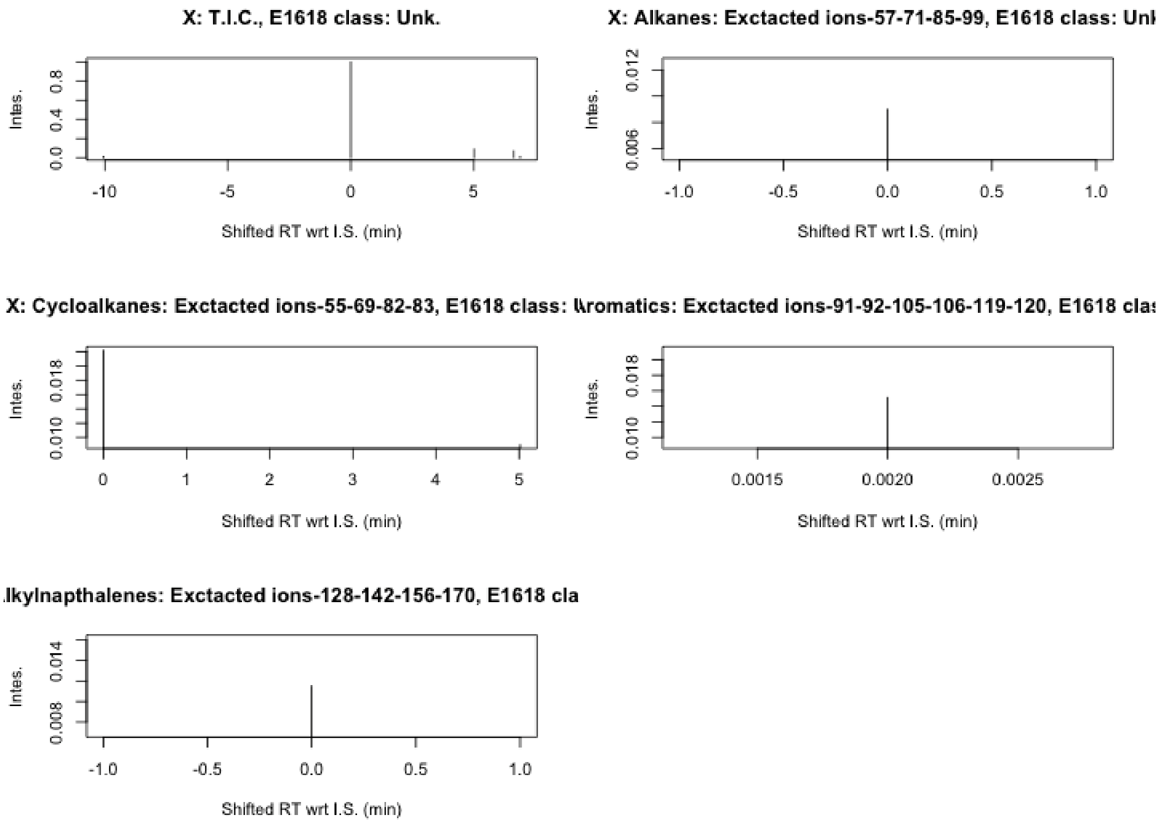
Unknown #	**Pred Classes**
38	HPD

Unknown 39: pw54



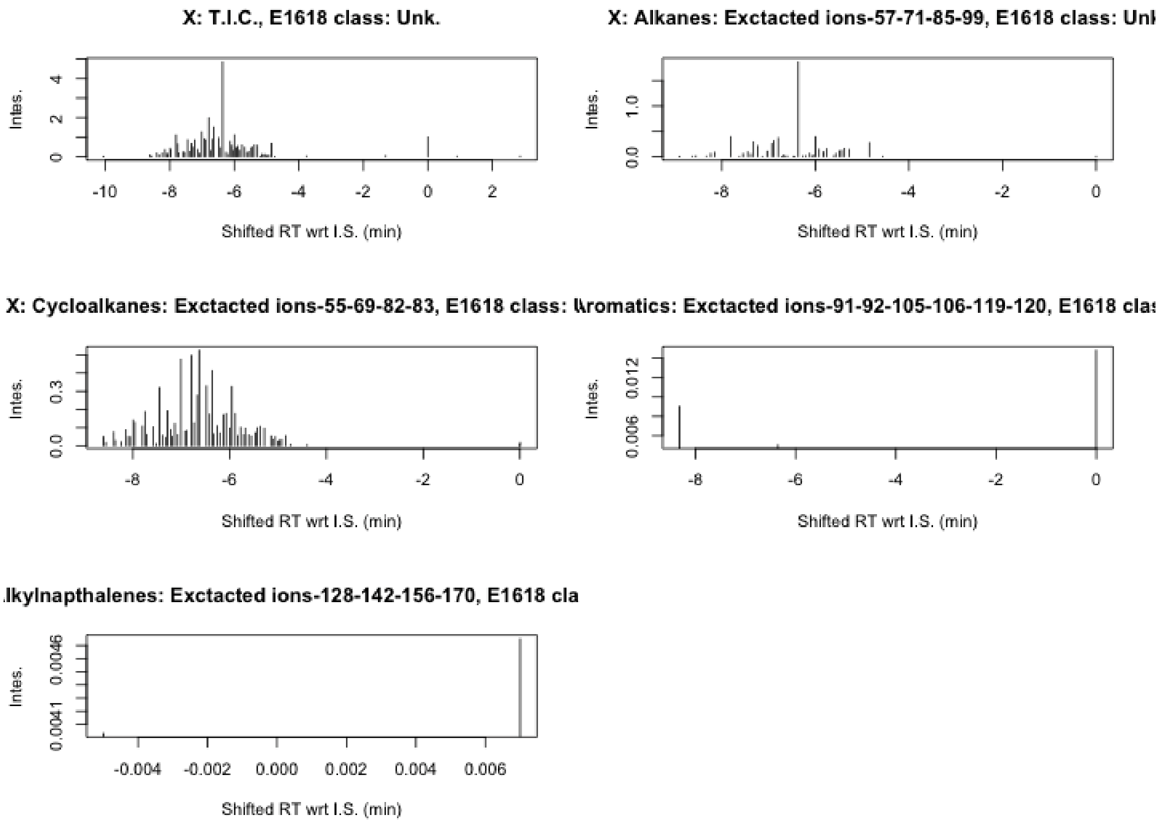
Unknown #	Pred Classes			Human Expert	Notes
39	MPD	M Oxy	M Misc	MPD	

Unknown 40: pw59



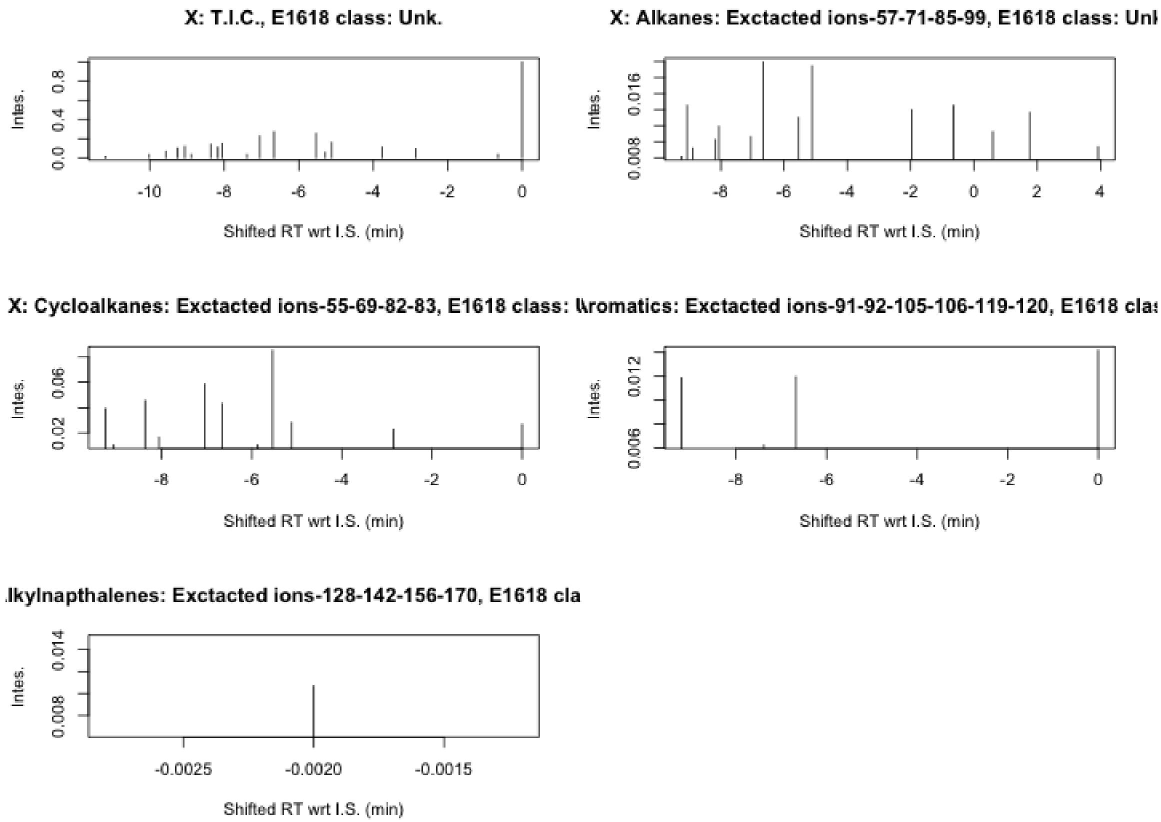
Unknown #	Pred Classes
40	H Oxy

Unknown 41: pw66



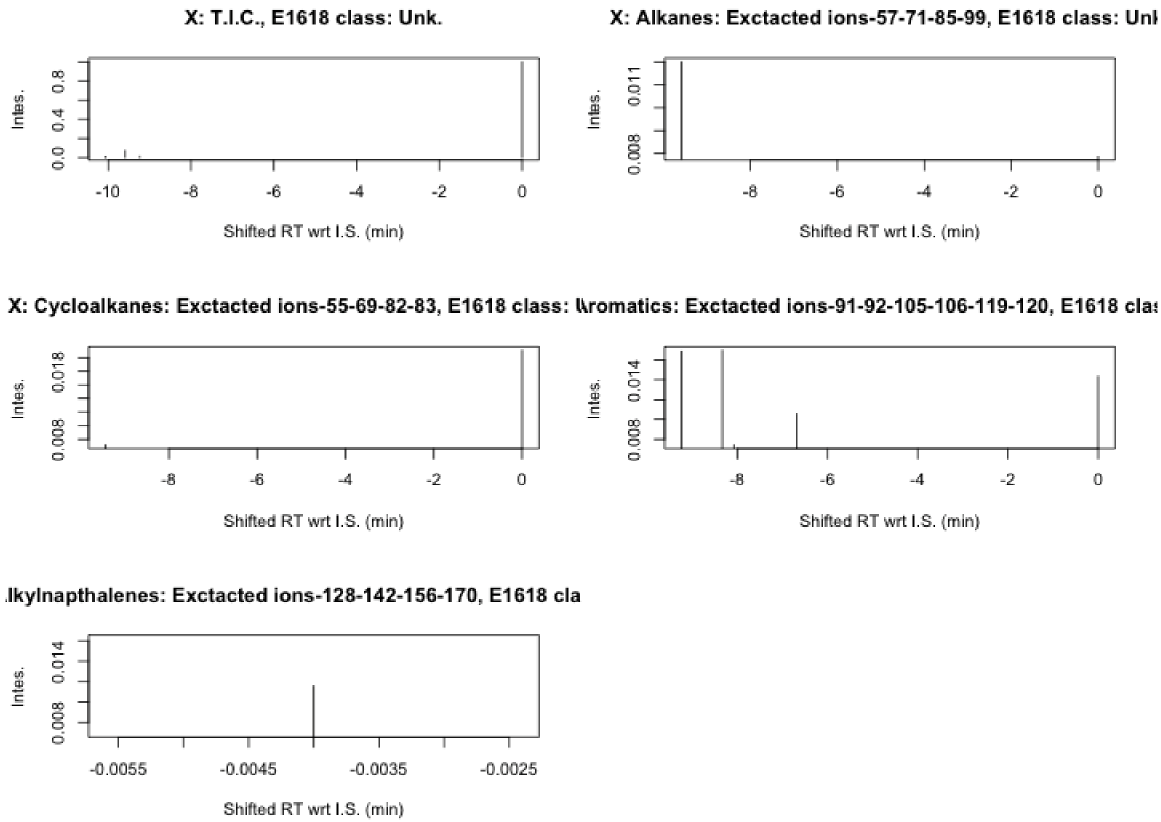
Unknown #	Pred Classes	
41	MPD	M Misc

Unknown 42: pwblk



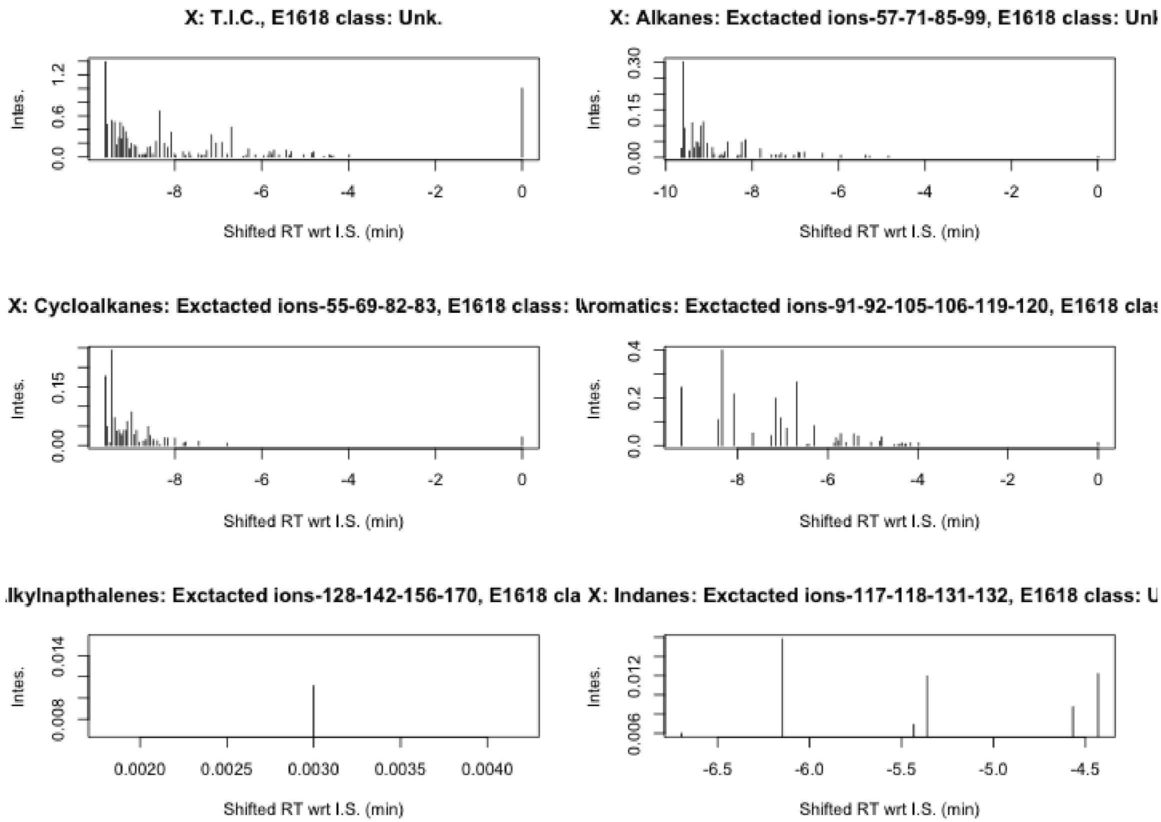
Unknown #	Pred Classes
42	CAN NOT ID

Unknown 43: pwe85



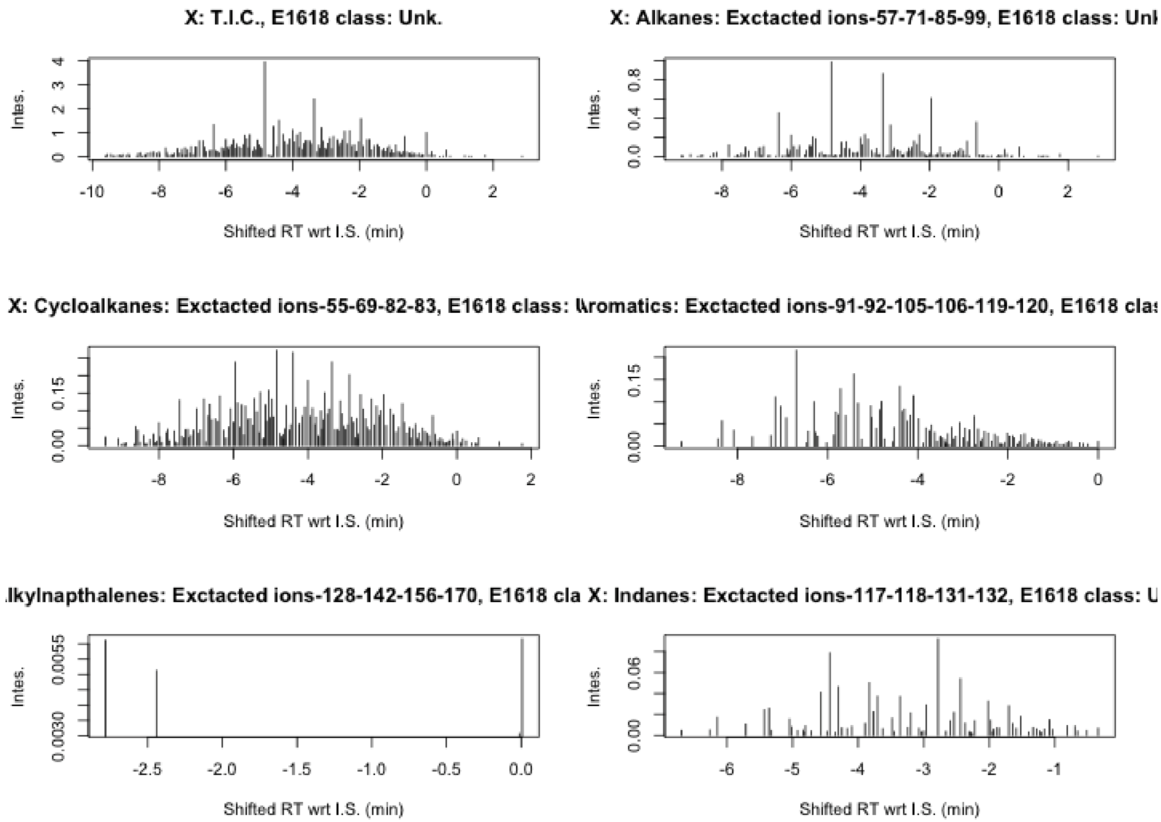
Unknown #	Pred Classes
43	H Oxy

Unknown 44: pwgas



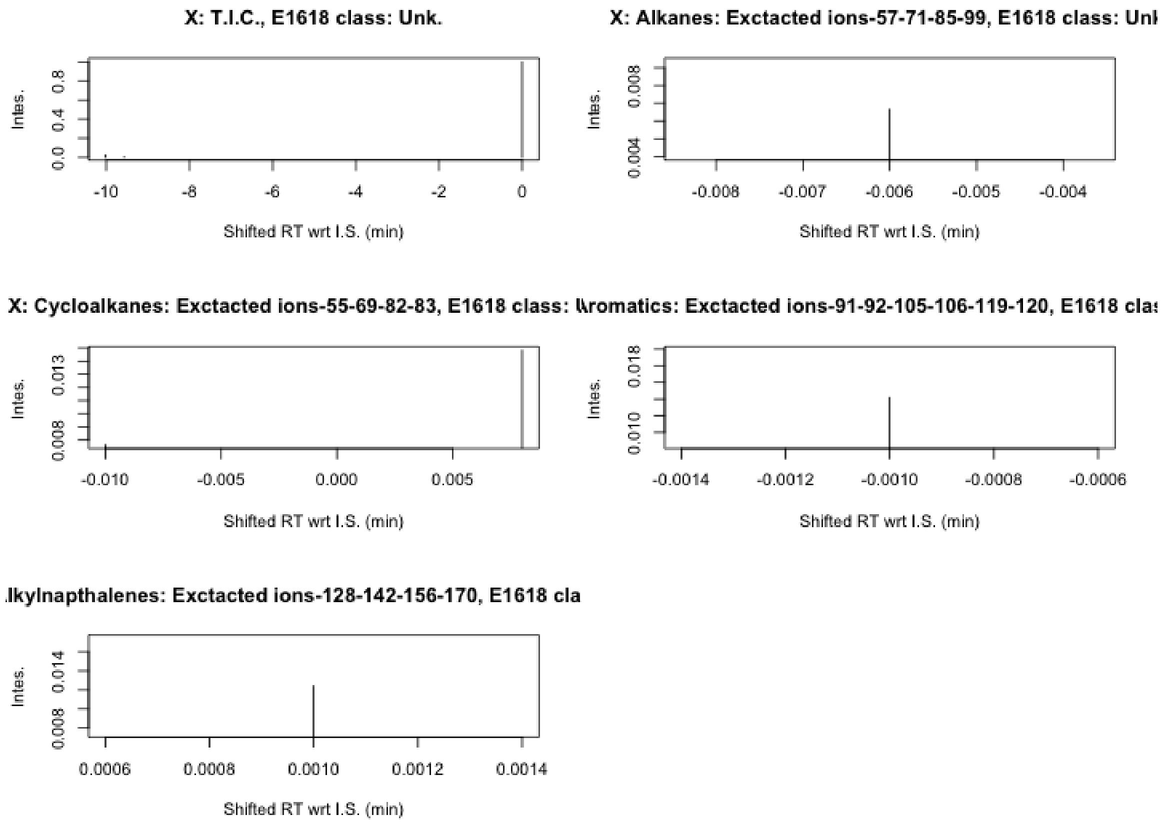
Unknown #	Pred Classes
44	M Misc

Unknown 45: pwker



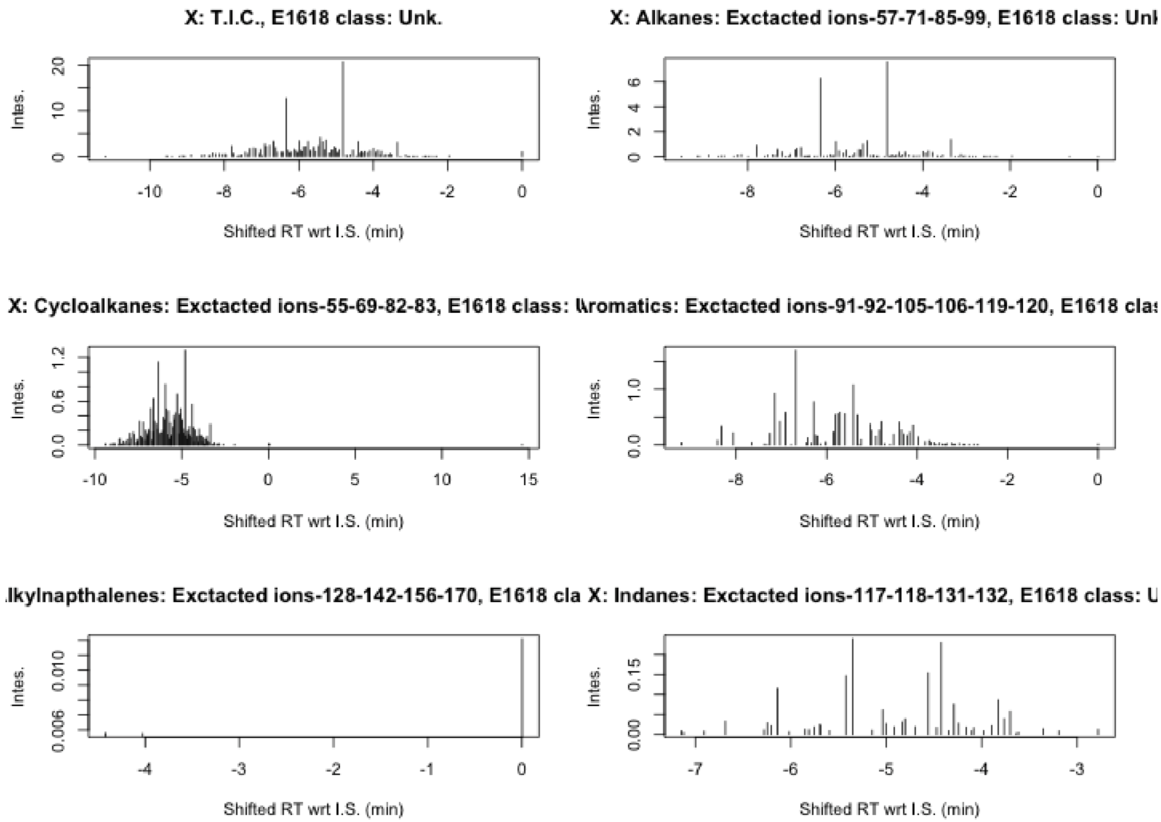
Unknown #	Pred Classes		
45	HPD	H Misc	H Oxy

Unknown 46: pwmtdb



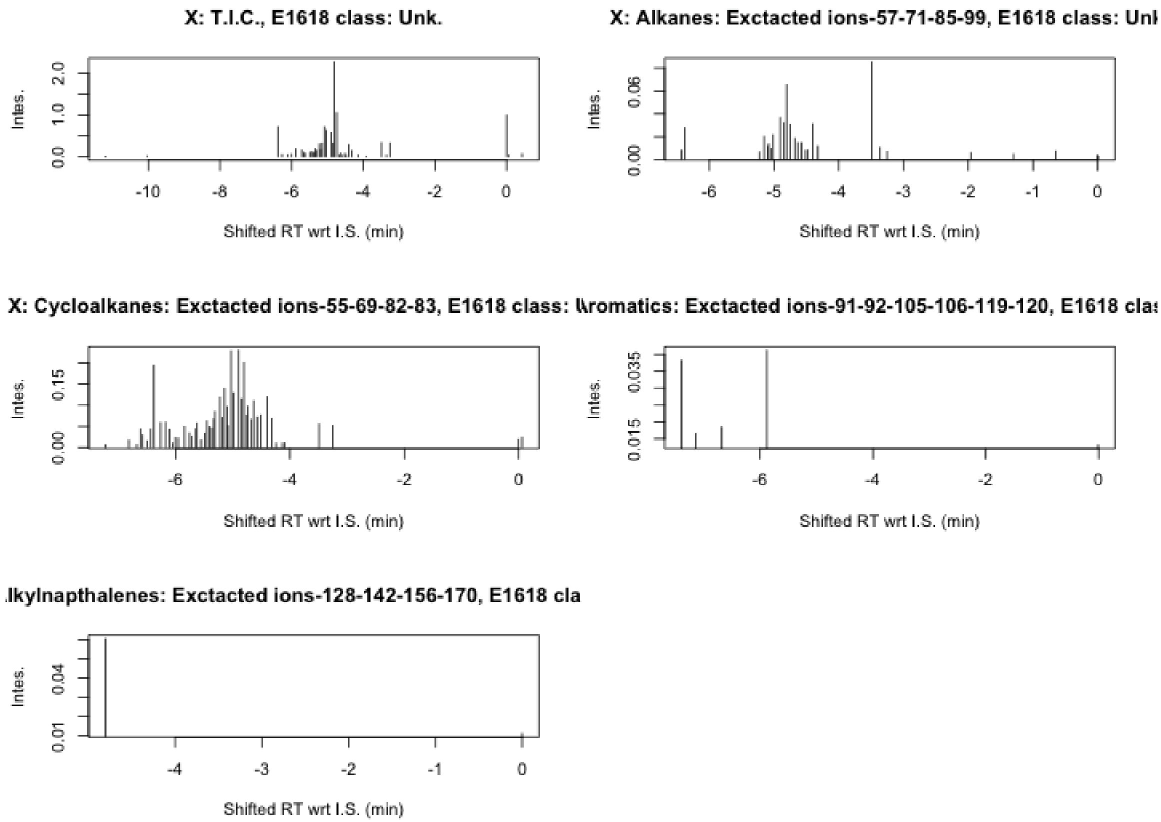
Unknown #	**Pred Classes**
46	H Oxy

Unknown 47: usbblc54



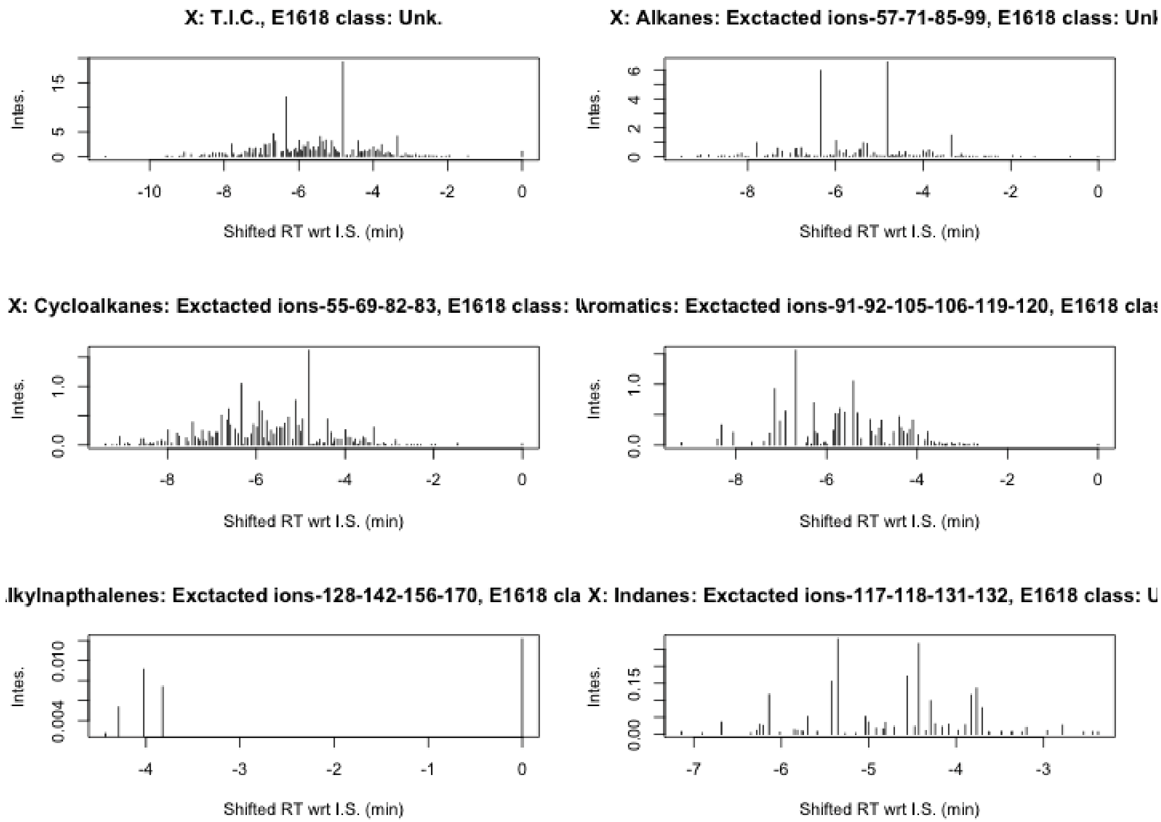
Unknown #	Pred Classes		
47	M Nap-par	M Oxy	
	MPD	M Misc	M Oxy

Unknown 48: usbblc59



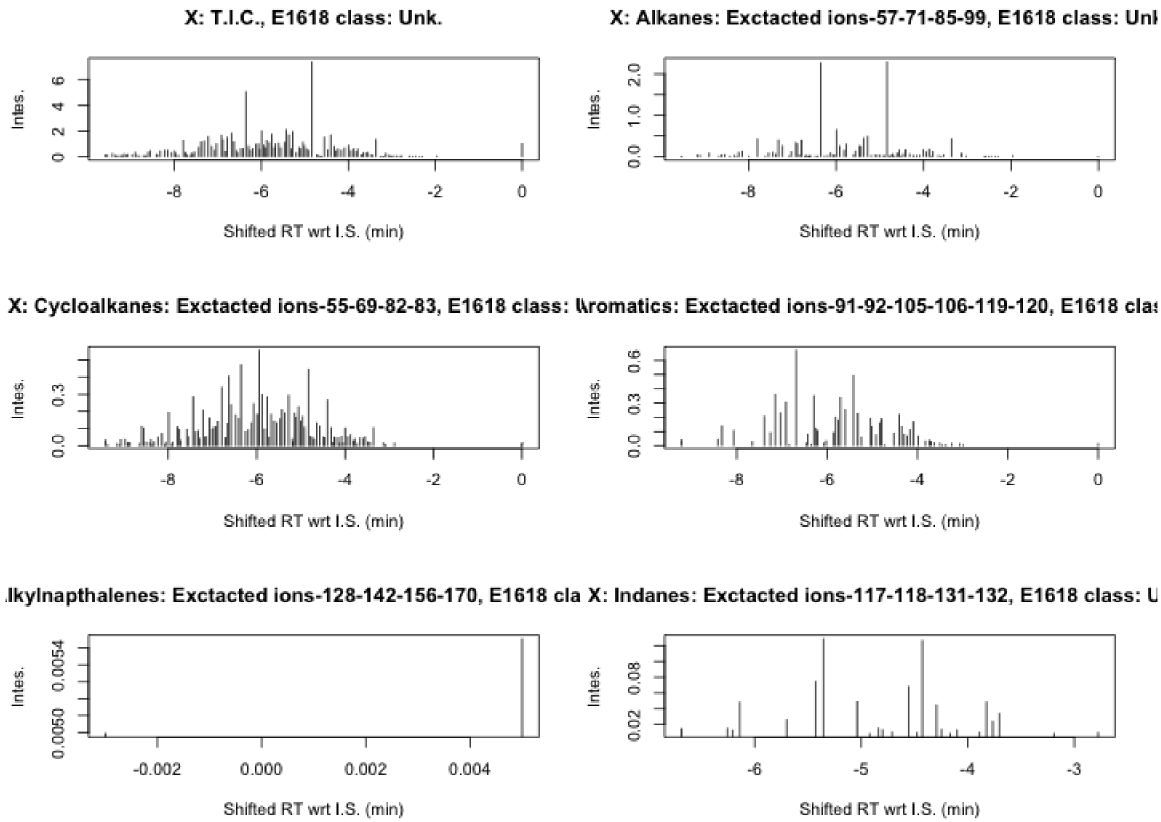
Unknown #	**Pred Classes**
48	CAN NOT ID

Unknown 49: ubsyp54



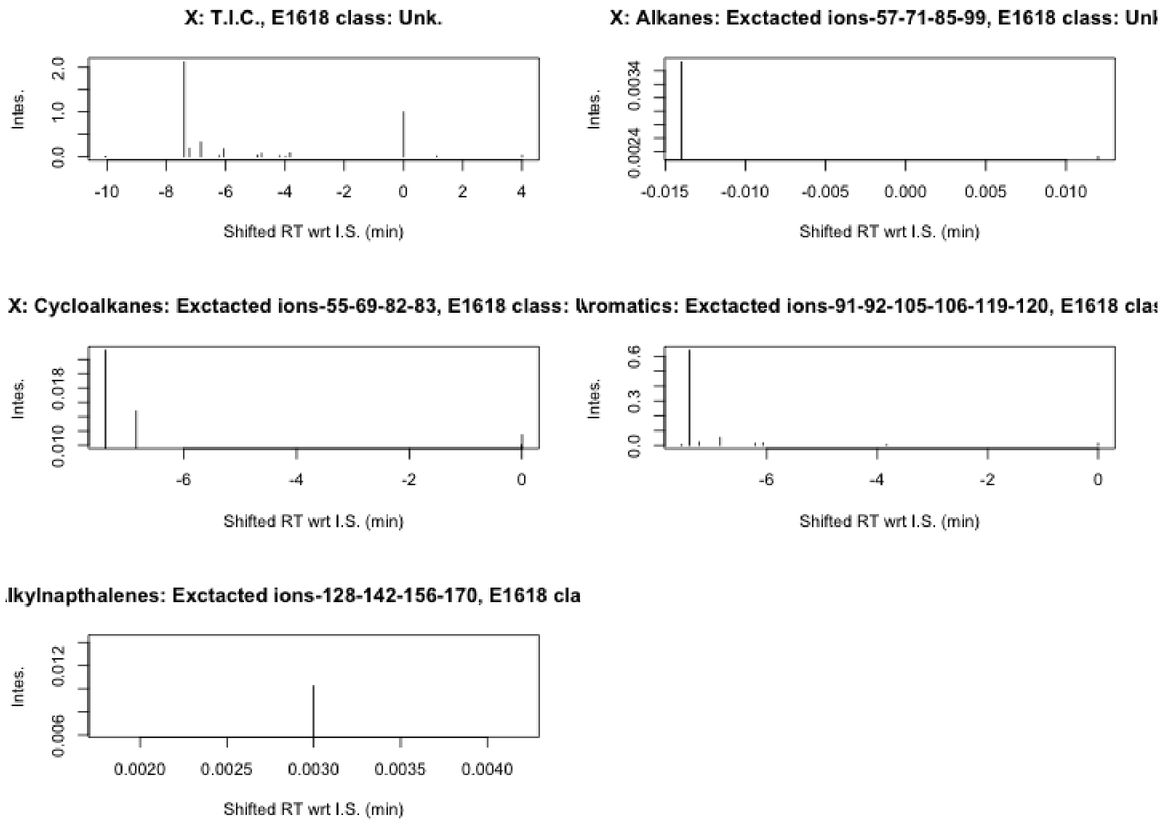
Unknown #	**Pred Classes**		
49	M Oxy	MPD	
	MPD	M Misc	M Oxy

Unknown 50: yp54



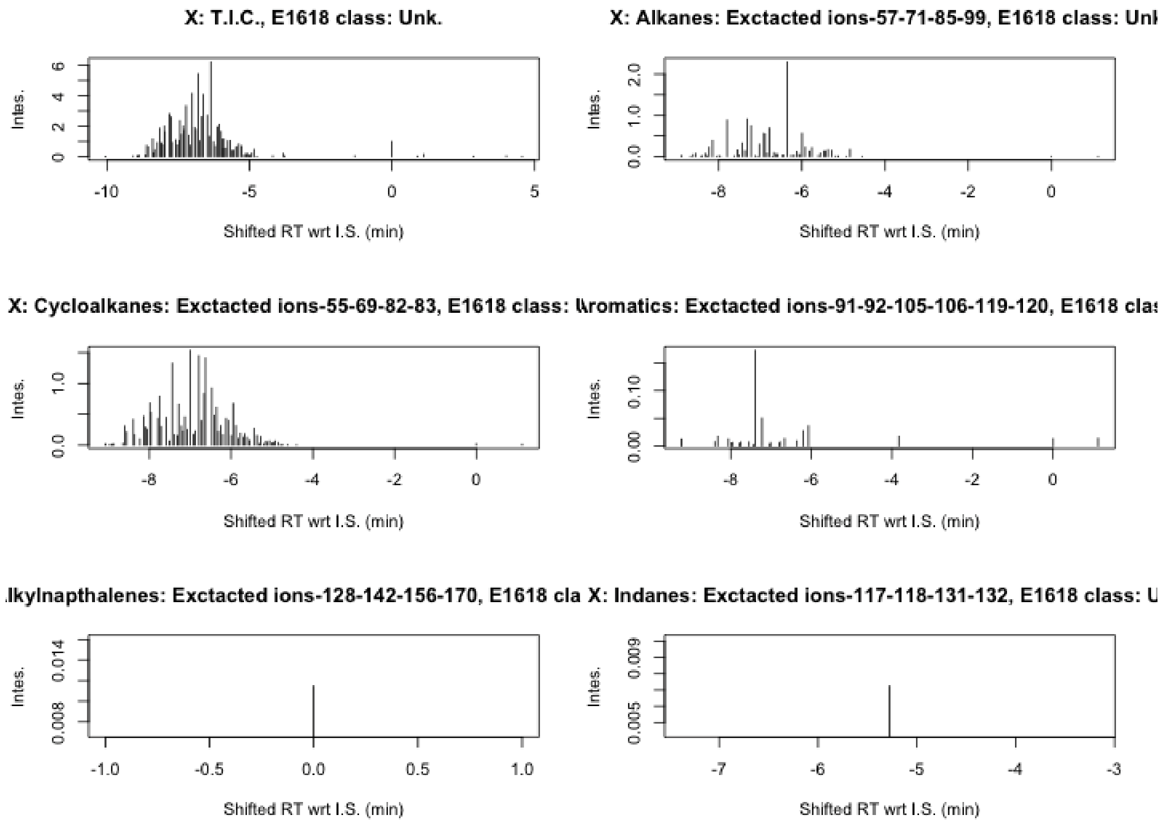
Unknown #	Pred Classes		
50	MPD	M Misc	M Oxy

Unknown 51: yp59



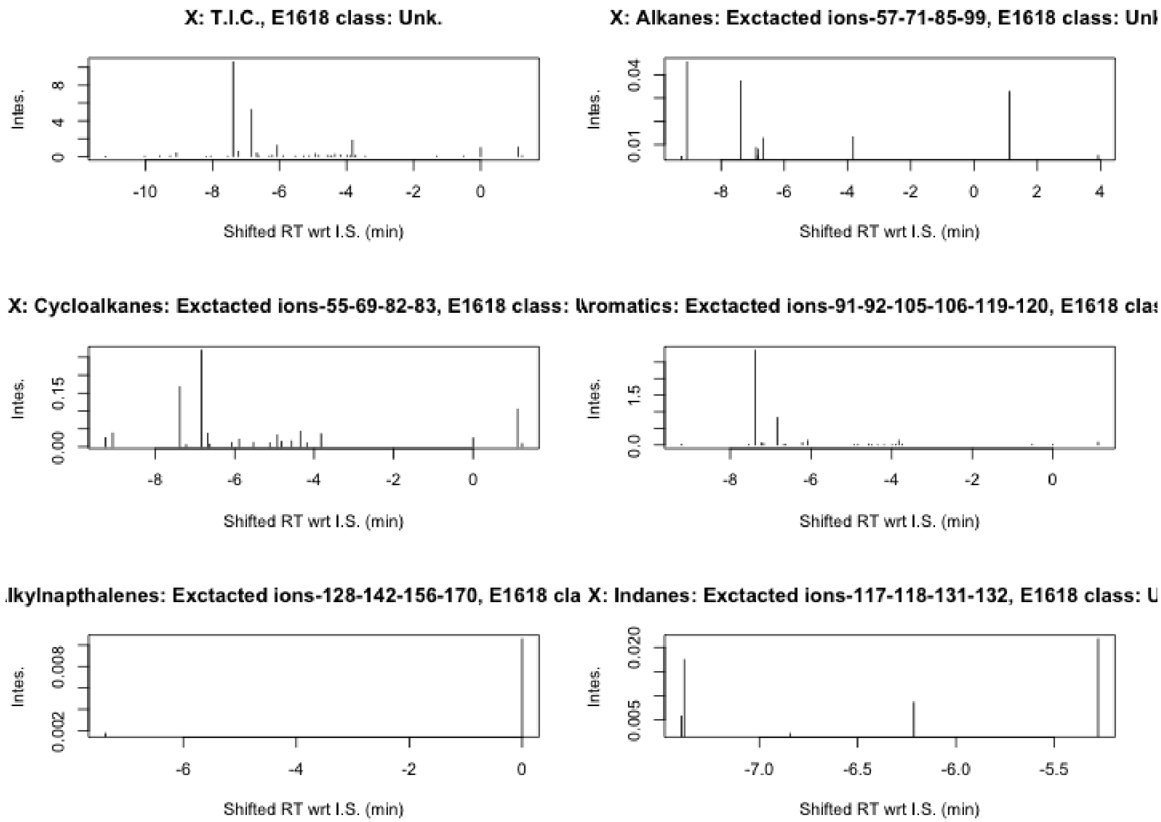
Unknown #	**Pred Classes**
51	H Aro

Unknown 52: yp66



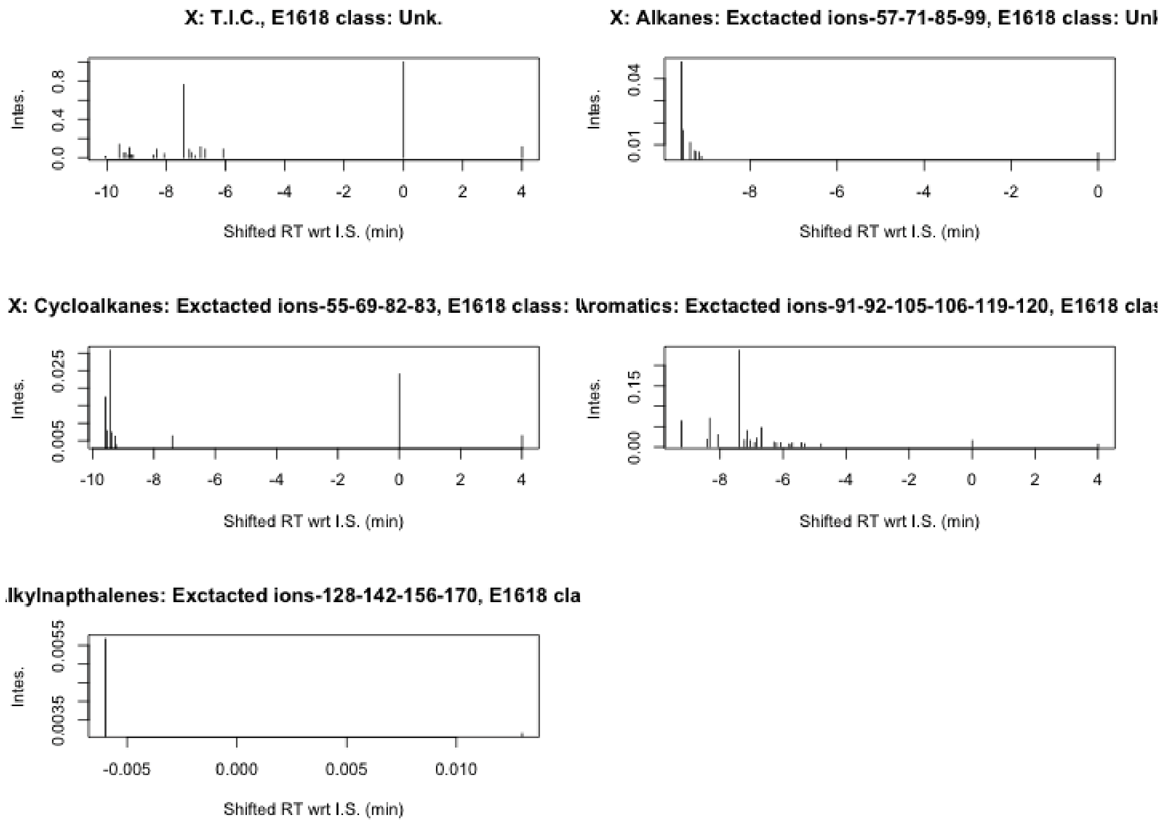
Unknown #	**Pred Classes**	
52	Med Oxy	MPD

Unknown 53: ypblk



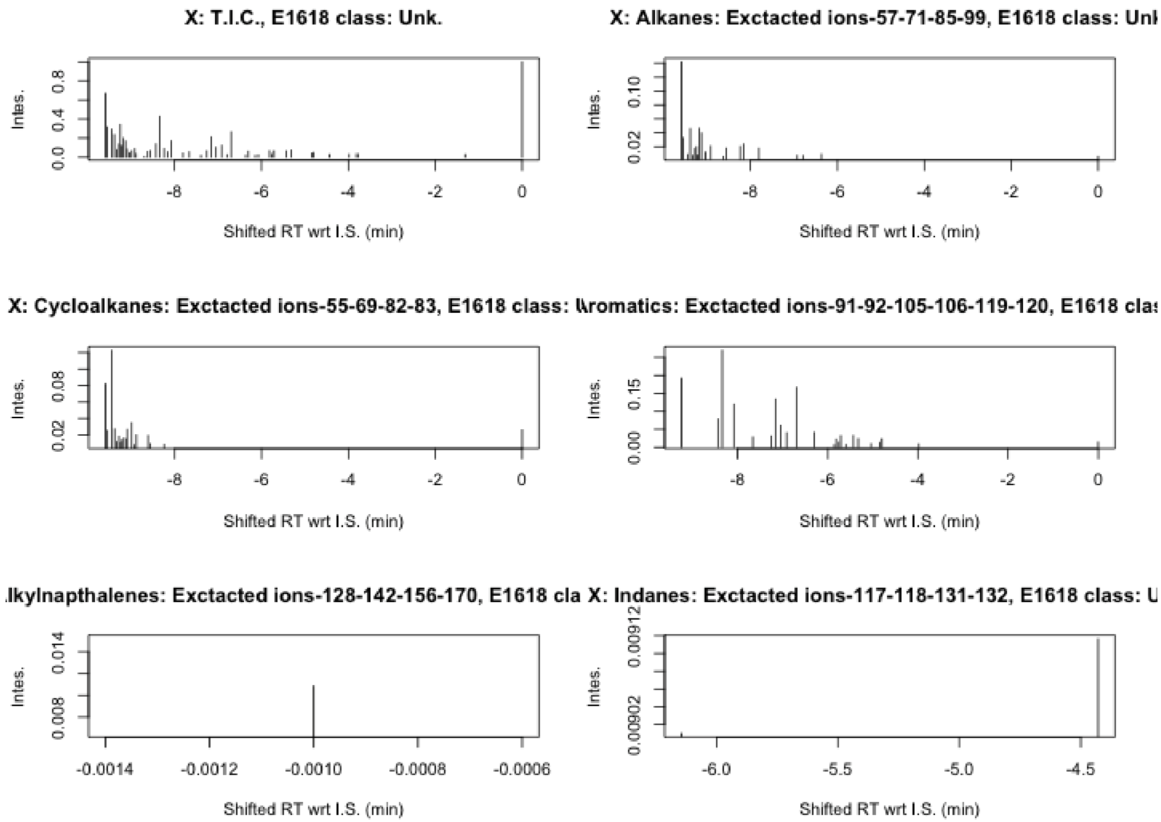
Unknown #	Pred Classes	
53	M Misc	Gasoline

Unknown 54: ype85



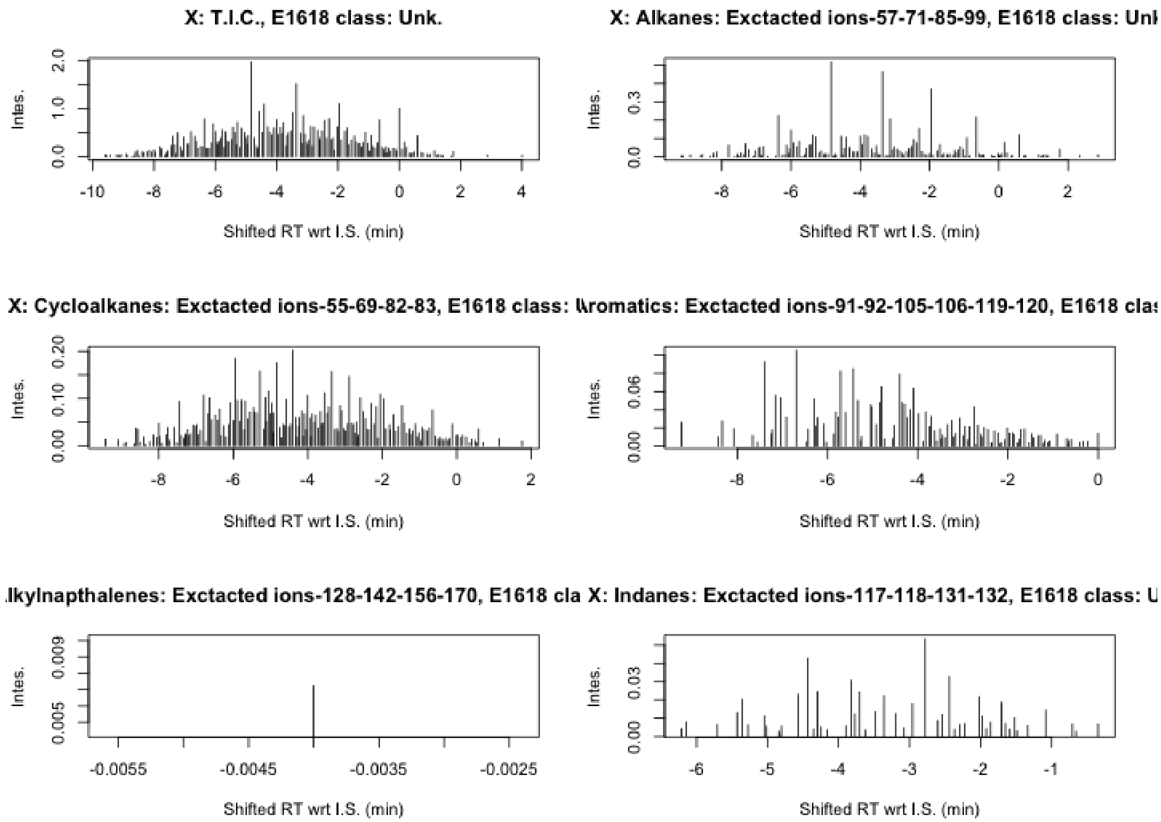
Unknown #	**Pred Classes**	
54	M Misc	M Oxy

Unknown 55: ypgas



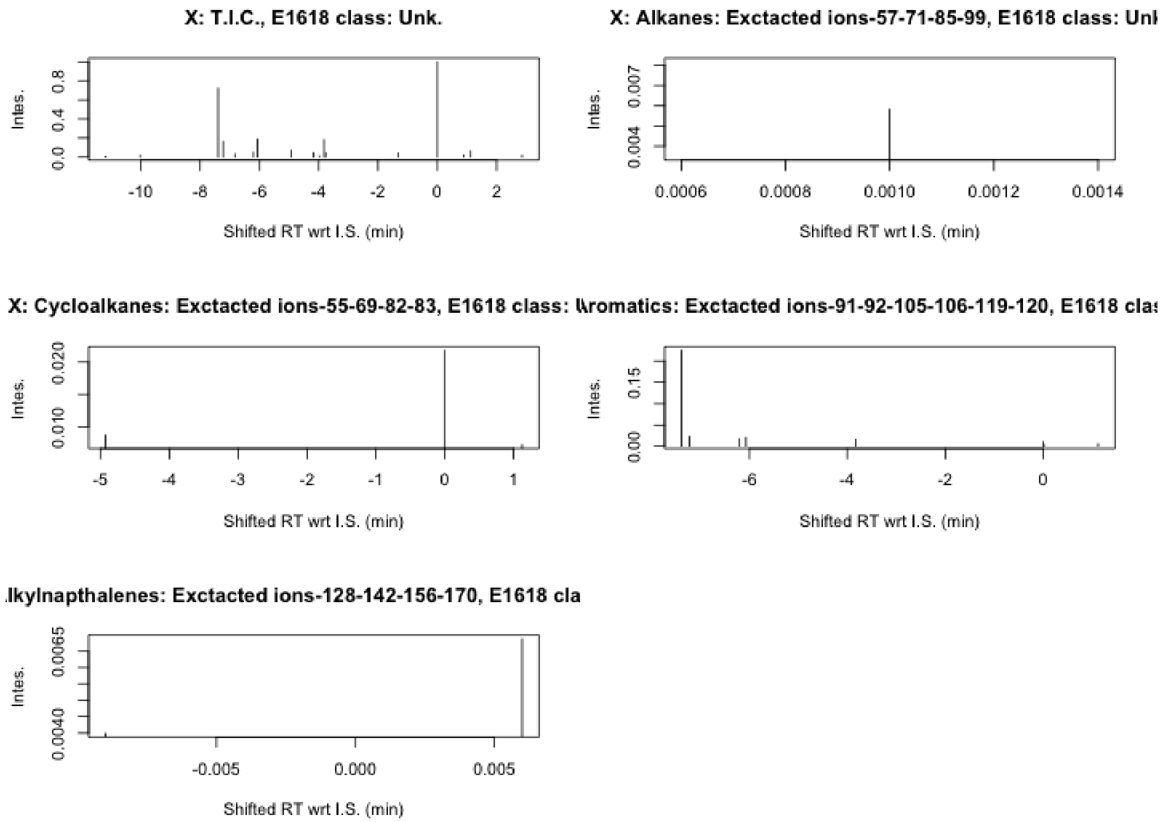
Unknown #	Pred Classes	
55	M Misc	M Oxy

Unknown 56: ypker



Unknown #	Pred Classes		
56	HPD	H Misc	H Oxy

Unknown 57: ypmtdb



Unknown #	Pred Classes
57	CAN NOT ID